Journal of Guidance, Control, and Dynamics, Volume 44, Issue 6, June 2021, pp. 1157-1182
DOI:10.2514/1.G004794

Robust On-Manifold Optimization for Uncooperative Space Relative Navigation with a Single Camera

Duarte Rondao[†]
Cranfield University, Defence Academy of the United Kingdom, SN6 8LA Shrivenham, United Kingdom

Nabil Aouf[‡]
City University of London, ECV1 0HB London, United Kingdom

Mark A. Richardson[§]
Cranfield University, Defence Academy of the United Kingdom, SN6 8LA Shrivenham, United Kingdom

Vincent Dubanchet[¶]
Thales Alenia Space, 06150 Cannes, France

Optical cameras are gaining popularity as the suitable sensor for relative navigation in space due to their attractive sizing, power and cost properties when compared to conventional flight hardware or costly laser-based systems. However, a camera cannot infer depth information on its own, which is often solved by introducing complementary sensors or a second camera. In this paper, an innovative model-based approach is demonstrated to estimate the six-dimensional pose of a target relative to the chaser spacecraft using solely a monocular setup. The observed facet of the target is tackled as a classification problem, where the three-dimensional shape is learned offline using Gaussian mixture modeling. The estimate is refined by minimizing two different robust loss functions based on local feature correspondences. The resulting pseudo-measurements are processed and fused with an extended Kalman filter. The entire optimization framework is designed to operate directly on the $SE(3)$ manifold, uncoupling the process and measurement models from the global attitude state representation. It is validated on realistic synthetic and laboratory datasets of a rendezvous trajectory with the complex spacecraft Envisat, demonstrating estimation of the relative pose with high accuracy over full tumbling motion. Further evaluation is performed on the open-source SPEED dataset.

Presented as paper 2018-2100 at the AIAA Guidance, Navigation, and Control Conference, Kissimmee, FL, 8–12 January 2018; submitted to AIAA Journal of Guidance, Control, and Dynamics 1 September 2019; first revision received 31 July 2020; second revision received 19 December 2020; accepted for publication 22 December 2020.

[†]PhD Candidate, Centre for Electronic Warfare, Information and Cyber.
[‡]Professor of Robotics and Autonomous Systems, Department of Electrical and Electronic Engineering.
[§]Professor of Electronic Warfare, Centre for Electronic Warfare, Information and Cyber.
[¶]R&D Engineer in AOCS and Robotics, CCPIF/AP-R&T.

I. Introduction

The concept of using optical sensors for spacecraft navigation has originated concurrently to the need for developing autonomous operations. The main rationale behind is twofold: the acquisition and processing of images is relatively simple enough to be self-contained on-board the spacecraft, avoiding the requirement of ground processing the data; and the same sensor feeds used for imaging detection and recognition can be used for navigation and mapping, alleviating additional sensors size and cost constraints. The maiden voyage of autonomous navigation was the Deep Space 1 (DS1) mission [1], which culminated in a fly-by with comet 19P/Borrelly in 2001. More recently, in 2014 the Rosetta mission [2] used optical navigation to rendezvous with the comet 67P/Churyumov–Gerasimenko. Historically, passive camera-based navigation has been reserved for orbit determination, cruise, and fly-by sequences. For proximity operations to small bodies, such as rendezvous, docking, or landing, the full relative pose is typically required. The difficulty of these scenarios is amplified when the target is uncooperative, meaning it does not relay direct information about its state. This is often overcome by resorting to more precise active sensors, such as Lidar, which have the advantage of supplying range information and being invariant to illumination changes [3, 4]. In addition to asteroids, Lidar has also been successfully used for relative navigation with artificial satellites [5] and developments in the field continue being made [6], but it remains challenging to integrate in on-board systems due to size and power constraints.

The motivation is therefore set to increase the technology readiness level of passive optical navigation through the development of supporting image processing (IP) techniques, in order to make it a viable, lower-costing alternative to active systems for the close range relative navigation problem. This is of paramount importance for proximity operations missions involving artificial satellites, especially in today's paradigm shift towards the democratization of space, where the number of spacecraft launches for the current decade is projected to reach a constant value of nearly 1000 per year and, by extension, the risk of collisions and overall orbital debris environment is only expected to intensify [7]. The European Space Agency's (ESA) e.Deorbit project, part of the CleanSpace initiative, is a proponent of this strategy to jump-start the adoption of active debris removal (ADR) techniques: an initial, smaller scale mission, e.Inspector, is envisaged to perform an in-orbit demonstration of IP algorithms aboard CubeSats to visually inspect the non-functioning Envisat spacecraft and infer its dynamical properties; the acquired data would then be used for validation purposes to ultimately capture and de-orbit the target [8, 9].

The most adopted approach to retrieve depth information in camera-based navigation systems involves the addition of a second camera forming a stereo setup: knowing the baseline between both, common landmarks detected in each image can be triangulated to obtain their relative distances to the camera frame [10]; still, adding a second camera increases not only the physical size of the system, but also the IP requirements. An alternative method is to measure the depth of the landmarks with a different sensor, or to initialize them based on a conjecture, such as information from the previous rendezvous stage (e.g. another sensor or inertial data) [11]; these are suitable when the distance to the target is large when compared to its dimensions, but the convergence of each landmark's depth to their true values becomes the

responsibility of the algorithm. A different technique, and the one adopted in this paper, consists in a model-based approach, which assumes that information about the three-dimensional structure of the target is known a priori. This structure can be decomposed into elementary landmarks ("features", as they are commonly called in the computer vision literature) that are annotated with 3D information. IP algorithms are then developed to solve the model-to-image registration problem, i.e. the coupled pose and correspondence problems, the latter which consists in establishing matches between the target's 3D structural information and the 2D features obtained by the camera and often overlooked in pure guidance, navigation and control (GNC) literature. In the circumstances where the target is artificial, such as in ADR, on-orbit servicing, or docking, it is justifiable to assume that its structure, or at least part of it, is known.

In this paper, a complete and innovative relative navigation framework using a monocular setup on the visible wavelength is proposed. The presented work follows a coarse-to-fine approach where a collection of training keyframes representing different facets of the target object are rendered offline using a 3D model of it. The method first determines the keyframe in the database closest to what the camera is imaging, producing a coarse estimate of the relative pose, which is then refined using local feature matching. This reduces the problem into a 2D-2D matching process, and shifts most of the computational burden to an offline training stage. Different hypotheses generated by the matching of features are fused with an extended Kalman filter (EKF), where the error state is defined to lie on the tangent space of the special Euclidean group $SE(3)$, providing a concise and elegant way to update the attitude using the exponential map. The prediction stage of the EKF is taken advantage of to help predict the locations of the features in the next frame, greatly improving the matching performance under adverse imaging conditions.

The contributions of the present paper are as follows: i) the tackling of the spacecraft pose estimation for relative navigation as a connected, funneled coarse classification to fine regression task; ii) the development of a spacecraft pose initialization method by modeling each viewpoint as a mixture of Gaussians to account for ambiguous shapes; iii) the introduction of a predictive feature matching technique to reduce the search space in estimation by detection, adding robustness to scenarios with tumbling and reflective targets where it would otherwise fail; iv) the synergistic integration of geometric pose estimation methods with a navigation filter via the proposed on-manifold optimization framework, where the measurement noise input of the latter is automatically computed as a byproduct of the former, with a consistent representation of the error-states. The spacecraft considered for simulations and also for experimental validations is Envisat, one of the few European Space Agency (ESA)-owned debris in low Earth orbit and a possible target of the e.Deorbit mission. Numerical simulations show that the coarse pose estimator achieves an accuracy of 90% for 20 deg bounds in azimuth and 92% for 20 deg in elevation, whereas the fine pose estimation algorithm yields an average error of 2.5% of range for the position and 1 deg for the attitude where the spacecraft undergoes complex tumbling motion. Additionally, the framework is benchmarked on the open-source Spacecraft PosE Estimation Dataset (SPEED), where the performance of the fine pose estimate on laboratory-acquired images is shown to be comparable to the winners of the 2019 Satellite Pose Estimation Challenge (SPEC).

Section II surveys the state-of-the-art for monocular camera-based relative navigation systems. Section III provides a review of the background theory used as the basis of this paper. Section IV presents a top-view outline of the developed framework. Section V illustrates the classifier designed to retrieve a coarse estimate of the relative pose. Section VI explains the motion estimation pipeline that runs nominally to generate fine estimates of the pose based on local feature matching. The EKF used for measurement fusion is presented in Section VII. Lastly, Section VIII showcases the results of the designed synthetic simulations and laboratory experiments, and Section IX presents the gathered conclusions.

II. Related Work

Most relative navigation methods using monocular cameras are based on sparse feature extraction: the input images are searched for two-dimensional features, typically interest points robust to transformations such as scale, perspective, and illumination changes, which are then tracked or matched with others to infer the relative camera pose (Section III). Such point features can be effectively obtained using detector-descriptor mechanisms such as SIFT [12], or the more recent ORB [13]; the current authors have recently benchmarked several of these state-of-the-art IP algorithms in the context of spacecraft relative navigation [14]. Nonetheless, the estimation of the relative pose is not limited to keypoints only: for instance, edges have notably been recognized early on as apt descriptors for the task [15, 16].

Regardless of the chosen feature type, approaches to pose estimation can be categorized into two main classes: model-free and model-based. Model-free methods do not require previous knowledge of the scene or target at hand, working by jointly estimating the camera's motion through the unknown environment and a mapping of it, in a process termed visual simultaneous localization and mapping (VSLAM) [17]. Recent work has shown VSLAM to be executable in real-time with good performance using keypoint detectors and tracking features from frame to frame [18] or even the raw image pixel intensities themselves [19]; it also been demonstrated to be applicable to relative navigation with an artificial target [20, 21]. However, a considerable disadvantage is that the scale of the estimated trajectory cannot be recovered.

Conversely, model-based approaches assume that information about the three-dimensional structure of the target is known a priori. In this case, IP algorithms are employed to solve the model-to-image registration problem, i.e. the coupled pose and correspondence problems, the latter which consists in establishing matches between the target's 3D structural information and the 2D features obtained by the camera and which is often overlooked in pure GNC literature. In the circumstances where the target is artificial, such as in ADR, on-orbit servicing, or docking, it is justifiable to assume that its structure, or at least part of it, is known. Within this approach, two paths can be followed [22]: tracking or detection. For the former, the system is initialised with a pose estimate and propagated by tracking features from frame to frame. A prominent technique in this category is virtual visual servoing (VVS) [23], which typically employs edge features of industrial computer aided design (CAD) models. It is assumed that the camera motion between frames is limited such that sampled 3D control points from this model are reprojected onto the image plane using an expected

pose accompanied by a one-dimensional scan to locate the corresponding edge on the feature space. Additional control points can be rendered as the found edge is subsequently tracked to the next frame. VVS has been successfully applied to spacecraft pose estimation first by Kelsey et al. [24]. Later on, Petit et al. [25] upgraded the VVS pipeline to include information from point and color features for tracking. A significant drawback of this design is that it relies on a graphics processing unit (GPU) for real-time rendering of the target model's depth map, making an implementation on current flight-ready hardware unlikely. Nevertheless, advances in model-based methods for the past decade have continued to focus on feature tracking either by disregarding the initialization stage [26] or by assuming that the chaser images the scene from a constant viewpoint [27–29], limiting their use for rendezvous with tumbling targets. On the other hand, model-based pose estimation by detection consists in matching 2D image features to a database of training features pre-computed offline. In the case of three-dimensional targets, this database is often obtained from a set of rendered viewpoints of the object, i.e. keyframes [30]. Despite the popularity of tracking methods for space applications, there have been proposals to apply detection methods to the problem: to the authors' best knowledge, it can be originally traced back to Cropp's doctoral thesis from 2001 [31] where, based on Dhome's [16] and Lowe's [32] work, pre-generated 3D edge features of a model of the target were matched to detected 2D image edges to retrieve the pose. Textureless features such as edges [33] or ellipses [34] gained popularity due to their robustness, but the matching process is often complex and relies on multiple hypotheses and long convergence times. This has recently shifted the focus towards point features, which can be efficiently matched using descriptors. The challenge in this case is in achieving robustness between test and train images [35], as these are often dissimilar in terms of baseline and illumination conditions; this has been tackled by in situ keypoint triangulation to shift the problem towards 3D descriptors [36, 37], or alternatively by combining keypoints with additional features for stability [38].

Regardless of the chosen model-based strategy, both benefit from initialization strategies for the incorporation of 3D information, either to propagate it the case of tracking, or to reduce the search space in the case of detection. In the computer vision literature, this has been treated as a coarse, or viewpoint-aware, object pose estimation. Traditional solutions worked by discretizing the object's 3D appearance according to a viewsphere and characterizing each bin according to its projected shape using moment invariants [39–41]. More recent methods follow a classification approach by clustering local features from each bin into a global representation combined with supervised techniques such as Bayesian classification [42] or support vector machines (SVMs) [43], or with unsupervised ones such as kernel density estimation (KDE) [44], to recover the viewpoint. Except for singular cases [45], initializers for spacecraft relative pose estimation have generally not taken advantage of such formulations, resorting instead to local features and either brute-force matching [46] or iterative methods [47]; despite simplifications to the search space [48, 49], these methods still rely on testing multiple hypothesis and discarding outliers, resulting in potentially long computation times due to the volume of features involved in the process.

Recently, deep learning methods, in particular convolutional neural networks (CNNs), have shown significant

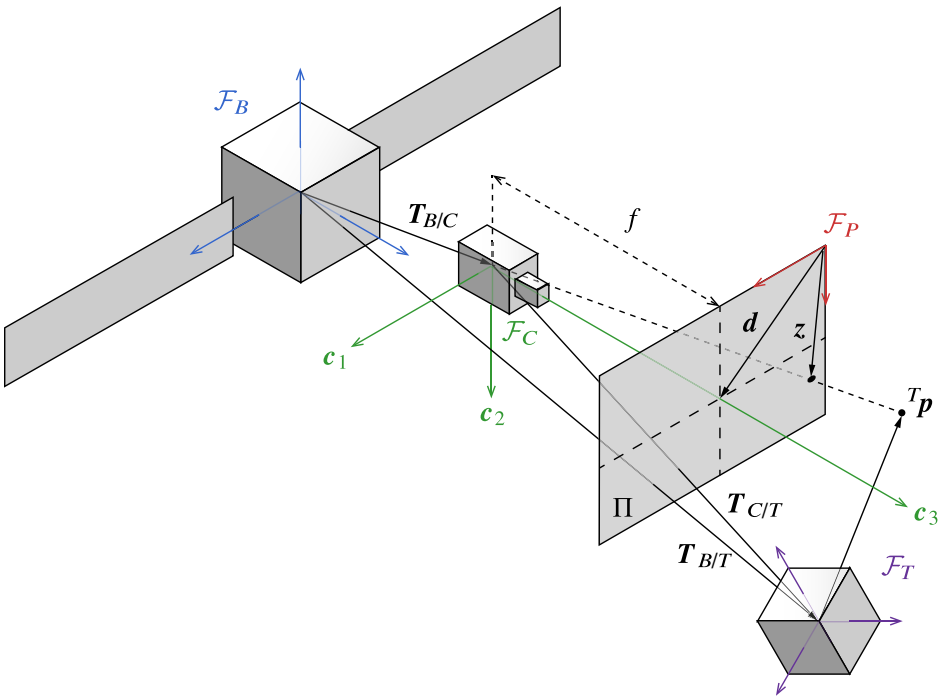


Fig. 1 Geometric relationships between frames of reference and landmark imaging.

improvements of the state-of-the-art for viewpoint classification [50, 51]. In particular, and concurrently to the initial drafting of this paper, CNN-based methods have begun to be adopted for the problem of spacecraft pose estimation [52, 53], fueled mainly by the ESA Advanced Concept Team's Satellite Pose Estimation Challenge (SPEC) [54]. These approaches are attractive as they allow for an end-to-end estimation of the relative pose by shifting the focus away from the feature modeling task. Interestingly, though, the winner of the competition trained a CNN for the sole detection of supervised keypoints on images of the target, relying then on geometrical techniques to solve for the relative pose [55], suggesting that pure deep-learning-based methods are still susceptible to be outperformed in this field. In addition, these bear some disadvantages such as large amounts of required training data and lower robustness to data outside the training regime. Pipelines relying on deep CNN backbones can also be characterized by large memory footprints and computational effort; the latter is typically tackled by deployment of the models to GPUs.

III. Mathematical Preliminaries

A. Camera Geometry

The relative pose estimation problem can be defined in terms of determining the rigid body transformation $\boldsymbol{T} = \boldsymbol{T}_{B/T}$ that links the frame of reference centered on the target object, $\mathcal{F}_\mathcal{T}$, to the chaser spacecraft's body frame, $\mathcal{F}_\mathcal{B}$. It is assumed that the chaser carries an on-board digital camera, which defines an additional frame of reference $\mathcal{F}_\mathcal{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3\}$, and that $\boldsymbol{T}_{B/C}$ is known. Figure 1 illustrates the different frames of reference used.

Let the origin of $\mathcal{F}_\mathcal{C}$ define the camera center of projection, or optical center, and let $\boldsymbol{c}_3$ be aligned with the sensor's boresight. To model the relationship between the three-dimensional scene and the two-dimensional image, a pinhole camera model is adopted, which assumes the projection of all rays through the common optical center [56]. Then, the scene is said to be projected on a plane Π perpendicular to $\boldsymbol{c}_3$ at a distance f to the optical center, i.e. the image plane, represented by the coordinate system $\mathcal{F}_\mathcal{P}$. Thus, a point $\boldsymbol{z}_P = \boldsymbol{z} = (z_1, z_2)^\top \in \Pi$ is obtained from a point in space $\boldsymbol{p}_T = (p_1, p_2, p_3)^\top_T \in \mathbb{R}^3$ in coordinates of frame $\mathcal{F}_\mathcal{T}$ (cf. Fig. 1) via perspective projection:

$$\boldsymbol{z} = \boldsymbol{\pi}\left(\boldsymbol{K}\boldsymbol{T} \otimes \boldsymbol{p}_T\right), \quad \text{with } \boldsymbol{K} \coloneqq \begin{bmatrix} f_x & 0 & d_1 \\ 0 & f_y & d_2 \\ 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

where f_x, f_y are the scalings of f by the sensor's dimensions and resolution in pixels, and $\boldsymbol{d} = (d_1, d_2)^\top$ are the coordinates of the principal point. The operator $\otimes$ is used to denote pose-point composition and general pose-pose composition. The matrix $\boldsymbol{K}$ represents the intrinsic parameters of the camera (in contrast to the extrinsic parameters, which are contained in $\boldsymbol{T}$), and can be obtained a priori through appropriate camera calibration. $\boldsymbol{\pi}(\tilde{\boldsymbol{z}}) \coloneqq \tilde{z}_3^{-1}(\tilde{z}_1, \tilde{z}_2)^\top$ is a projective function that applies the mapping from the 2D projective space $\mathbb{P}^2$ to $\mathbb{R}^2$ on a point expressed in homogeneous coordinates. Note that the equivalence $\tilde{\boldsymbol{z}} = \lambda\tilde{\boldsymbol{z}}$ exists for any $\lambda \in \mathbb{R} \setminus \{0\}$. For simplicity, the tilde $(\tilde{\cdot})$ notation for homogeneous points is dropped whenever the involved dimensions are unambiguous. Equation (1) shows that the depth of a 3D point is lost after projection.

B. Lie Groups

The rigid body transformation matrix $\boldsymbol{T}$ is the homogeneous representation of an element of the 3-dimensional special Euclidean group [57]:

$$SE(3) \coloneqq \left\{ \boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}_{1\times3} & 1 \end{bmatrix} \;\middle|\; \boldsymbol{R} \in SO(3),\, \boldsymbol{t} \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times4}. \tag{2}$$

$SE(3)$ is a 6-dimensional smooth manifold (concretely, a Lie group, $\mathcal{G}$) with matrix multiplication as the group operation. Note that $SE(3)$ (or, analogously, $SO(3)$) is not a vector space. This means that the sum of two transformation (resp. rotation) matrices is not a valid transformation (resp. rotation) matrix. Since optimization frameworks are usually designed for corrective steps that consist in the addition of Euclidean spaces, incorporating a pose (resp. a rotation) is not a direct task.

However, one can exploit the local Euclidean structure of a manifold $\mathcal{M}$, i.e. the tangent space at each point $x \in \mathcal{M}$, $T_x\mathcal{M}$. The tangent space of a Lie group $\mathcal{G}$ at the identity, $T_I\mathcal{G}$, is the Lie algebra, which is a vector space [58]. The Lie algebra therefore linearizes the Lie group near the identity element while conserving its structure [58, 59].

The retraction mapping $T_I\mathcal{G} \to \mathcal{G}$ is the exponential map, and for matrix Lie groups it corresponds to matrix exponentiation:

$$\exp(\boldsymbol{X}) = \sum_{k=0}^{\infty} \frac{1}{k!} \boldsymbol{X}^k, \quad \boldsymbol{X} \in \mathbb{R}^{n \times n}. \tag{3}$$

The $(\cdot)^\wedge$ operator* is used to map a vector $\boldsymbol{\phi} \in \mathbb{R}^3$ to the Lie algebra of $SO(3)$:

$$(\cdot)^\wedge_{\mathfrak{so}(3)} \colon \mathbb{R}^3 \to \mathfrak{so}(3), \quad \boldsymbol{\phi}^\wedge := \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix}^\wedge \mapsto \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix}. \tag{4}$$

This is frequently found in the literature with the analogous representation $(\cdot)^\times$ since the mapping yields a 3×3 skew-symmetric matrix such that $\boldsymbol{a} \times \boldsymbol{b} = \boldsymbol{a}^\times \boldsymbol{b}$. The inverse mapping $\mathfrak{so}(3) \to \mathbb{R}^3$ is performed with the $(\cdot)^\vee$ operator. These two operators are overloaded to achieve a mapping between $\mathbb{R}^6$ and the Lie algebra of $SE(3)$:

$$(\cdot)^\wedge_{\mathfrak{se}(3)} \colon \mathbb{R}^6 \to \mathfrak{se}(3), \quad \boldsymbol{\xi}^\wedge := \begin{pmatrix} \boldsymbol{\rho} \\ \boldsymbol{\phi} \end{pmatrix}^\wedge \mapsto \begin{bmatrix} \boldsymbol{\phi}^\wedge & \boldsymbol{\rho} \\ \boldsymbol{0}_{1\times 3} & 0 \end{bmatrix} \quad \text{with} \quad \boldsymbol{\rho}, \boldsymbol{\phi} \in \mathbb{R}^3. \tag{5}$$

For $SO(3)$ and $SE(3)$, Eq. (3) has a known closed form expression [57]:

$$\exp_{SE(3)} \colon \mathfrak{se}(3) \to SE(3), \quad \boldsymbol{\xi}^\wedge \mapsto \begin{bmatrix} \exp_{SO(3)}(\boldsymbol{\phi}^\wedge) & \boldsymbol{N}(\boldsymbol{\phi})\boldsymbol{\rho} \\ \boldsymbol{0}_{1\times 3} & 1 \end{bmatrix}, \tag{6}$$

with $\exp_{SO(3)}(\cdot)$ given by the Rodrigues rotation formula and $\boldsymbol{N}(\boldsymbol{\phi}) := \boldsymbol{I}_3 + (1 - \cos\|\boldsymbol{\phi}\|)\boldsymbol{\phi}^\wedge/\|\boldsymbol{\phi}\|^2 + (\|\boldsymbol{\phi}\| - \sin\|\boldsymbol{\phi}\|)\boldsymbol{\phi}^{\wedge 2}/\|\boldsymbol{\phi}\|^3$.

It is occasionally convenient to use the adjoint action of a Lie group on its Lie algebra [60]. For $SE(3)$:

$$\mathrm{Ad}_{SE(3)} \colon SE(3) \to \mathbb{R}^{6\times 6}, \quad \boldsymbol{T} \mapsto \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t}^\wedge \boldsymbol{R} \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{R} \end{bmatrix}. \tag{7}$$

Let $g \in \mathcal{G}$. If $\boldsymbol{T} = \boldsymbol{T}(g)$ is the homogeneous representation of the group element g, then $\boldsymbol{\xi'}^\wedge = \boldsymbol{T}\boldsymbol{\xi}^\wedge\boldsymbol{T}^{-1}$ also yields an element of $\mathfrak{se}(3)$ and the relation can be written linearly in $\mathbb{R}^6$ as $\boldsymbol{\xi'} = \mathrm{Ad}(\boldsymbol{T})\boldsymbol{\xi}$. Furthermore, the adjoint action of the Lie algebra on itself is

$$\mathrm{ad}_{\mathfrak{se}(3)} \colon \mathfrak{se}(3) \to \mathbb{R}^{6\times 6}, \quad \boldsymbol{\xi}^\wedge \mapsto \begin{bmatrix} \boldsymbol{\phi}^\wedge & \boldsymbol{\rho}^\wedge \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{\phi}^\wedge \end{bmatrix}, \tag{8}$$

such that the expression for the Lie bracket of $\mathfrak{se}(3)$ can be written as $[\boldsymbol{\xi}_0, \boldsymbol{\xi}_1] := \boldsymbol{\xi}_0\boldsymbol{\xi}_1 - \boldsymbol{\xi}_1\boldsymbol{\xi}_0 = (\mathrm{ad}(\boldsymbol{\xi}_0^\wedge)\boldsymbol{\xi}_1)^\wedge$.

C. Optimization Framework for Manifolds

The manifold optimization framework developed for this work revolves around the importance of the tangent space as a local vector space approximation for the pose manifold. It is well-known that, for manifolds endowed with a Riemannian metric, retractions (approximations of the exponential map accurate up to first order) are gradient-preserving

*Not to be confused with $\hat{\cdot} \neq (\cdot)^\wedge$.

[61]. This means that optimization problems based on Euclidean spaces relying on the computation of gradients (or some approximation thereof) can be generalized to (nonlinear) manifolds via retraction mappings. By extension, since a Lie group is a smooth manifold, any $\mathcal{G}$ such as $SE(3)$ can be endowed with a Riemannian metric [62]. As such, the exponential map can be used as the bridge to locally convert an optimization problem stated in terms of $\boldsymbol{T}$ to the more tractable vector space of the corresponding Lie algebra element $\boldsymbol{\xi}^\wedge$ (or simply its compact representation $\boldsymbol{\xi} \in \mathbb{R}^6$), where methods of Euclidean analysis can be used. Formally, for an incremental correction $\boldsymbol{\xi}$, a solution in the manifold can be propagated as

$$\boldsymbol{T}' = \exp\left(\boldsymbol{\xi}^\wedge\right)\boldsymbol{T}, \quad \text{with} \quad \boldsymbol{T}', \boldsymbol{T} \in SE(3) \quad \text{and} \quad \boldsymbol{\xi}^\wedge \in \mathfrak{se}(3), \tag{9}$$

where the left-product convention has been adopted. It is also useful to see $SE(3)$ as a semi-direct product of manifolds $SO(3) \rtimes \mathbb{R}^3$, as one might be interested in working with isomorphic representations of $SO(3)$, such as the special unitary group $SU(2)$ of unit quaternions, with the well-known isomorphism [63]:

$$\boldsymbol{R}(\boldsymbol{q}) = \left(q^2 - \|\boldsymbol{e}\|^2\right)\boldsymbol{I}_3 - 2q\boldsymbol{e}^\wedge + 2\boldsymbol{e}\boldsymbol{e}^\top, \quad \text{with} \quad \boldsymbol{q} \in SU(2) \quad \text{and} \quad \boldsymbol{R} \in SO(3), \tag{10}$$

where $\boldsymbol{e}$ and q are the vector and scalar parts of the quaternion, respectively, which is written as

$$\boldsymbol{q} \coloneqq \begin{pmatrix} \boldsymbol{e} \\ q \end{pmatrix} \tag{11}$$

As it is familiar in the space domain, the composition of two attitude quaternions is taken in the form of Shuster's product [64], meaning that rotations are composed in the same order as for rotation matrices:

$$\boldsymbol{q}_0 \otimes \boldsymbol{q}_1 = \begin{bmatrix} q_0\boldsymbol{I}_3 - \boldsymbol{e}_0^\wedge & \boldsymbol{e}_0 \\ -\boldsymbol{e}_0^\top & q_0 \end{bmatrix}\boldsymbol{q}_1. \tag{12}$$

If a Lie group $\mathcal{G}$ is a manifold obtained through the semi-direct product of some isomorphism of $SO(3)$ and $\mathbb{R}^3$, then $\mathcal{G}$ is isomorphic to $SE(3)$ as a manifold, but not as a group [59]. The operator $\oplus\colon \mathcal{G} \times \mathbb{R}^6 \to \mathcal{G}$ is thus defined to generalize a composition of a group element $g \in \mathcal{G}$ representing a pose and an element $\boldsymbol{\xi}$ which is the compact representation in $\mathbb{R}^6$ of $\boldsymbol{\xi}^\wedge \in \mathfrak{se}(3)$:

$$g' = g \oplus \boldsymbol{\xi}, \quad \text{with} \quad g, g' \in \mathcal{G}, \quad \text{and} \quad \mathcal{G} \cong SE(3). \tag{13}$$

Likewise, one defines the inverse operation $\ominus\colon \mathcal{G} \times \mathcal{G} \to \mathbb{R}^6$ that yields the compact representation of an element of the Lie algebra.

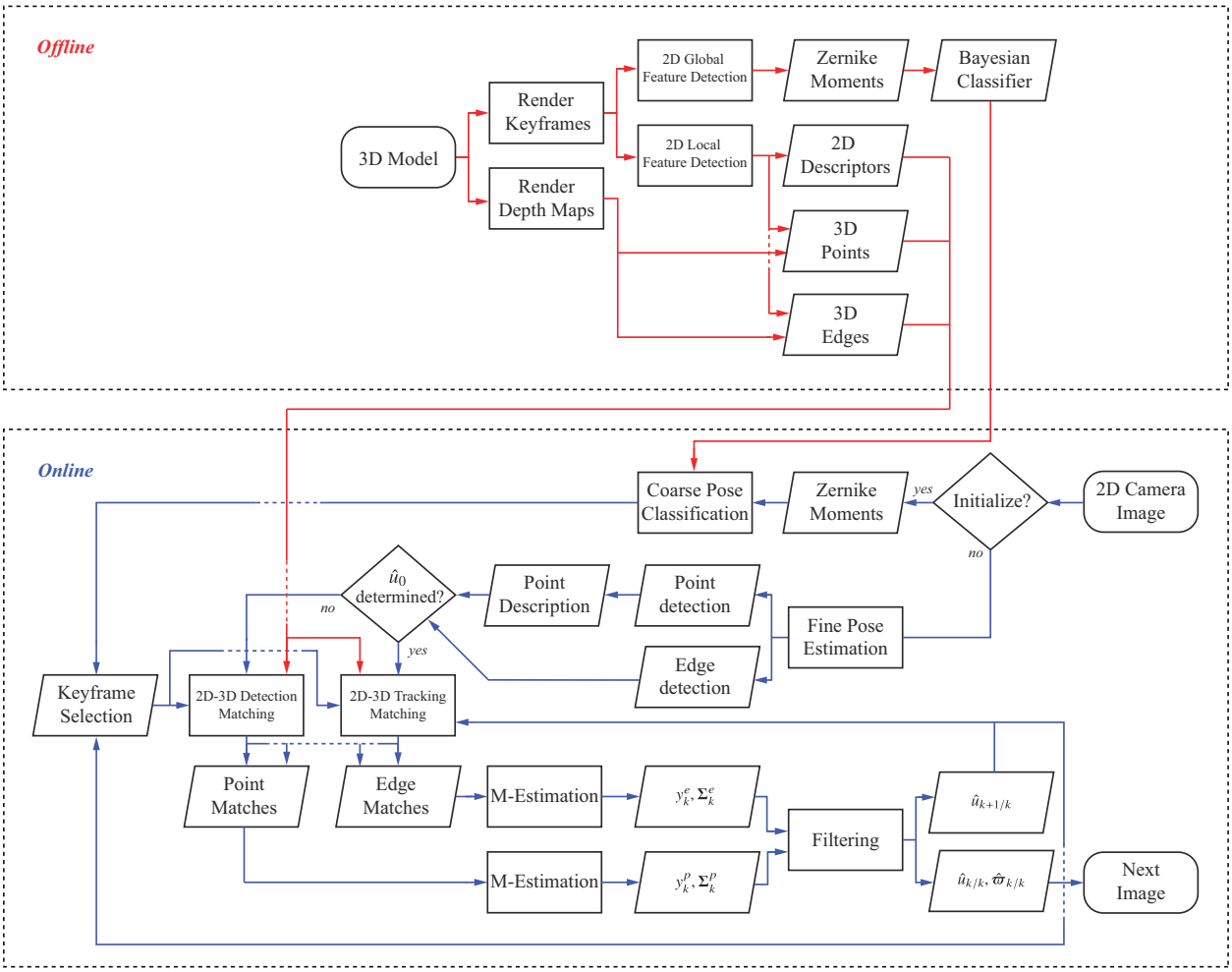


Fig. 2 Simplified flowchart of the relative navigation framework's structure.

IV. System Overview

The architecture of the proposed relative navigation framework can be divided into two fundamental components: an offline training stage and an online pipeline.

The offline training stage has as its objective to discretize, categorize, and represent the three-dimensional structure of the target so that it can be utilized in the two-dimensional environment of the online stage. Two sets of images are sampled from different viewing angles of the target's CAD model. The first is a set of keyframes, each of which contains textural information from the target as imaged from that viewpoint. The second is a set of depth maps, each of which has the same scene structure as its corresponding keyframe, but the value of each pixel represents the distance of that point in the target to the image plane. For each keyframe, the shape of the target is mathematically represented using complex Zernike moment (ZM). The distribution of the ZM feature vector elements per class is modeled using Gaussian mixture models (GMMs), which will define the likelihood probabilities in the training of a Bayesian classifier later employed to match the target's facet as observed by the on-board camera to the closest keyframe in the database, defining the coarse

pose classification module (Section V).

The keyframes are also processed with a feature point detector. The aim is to identify keypoints distinguishable enough to be matched to the same keypoint in the context of the online pipeline. Each keypoint is subjected to a feature descriptor, which generates a signature vector used to search and match in the descriptor space. Using the depth map corresponding to its keyframe, each keypoint is annotated with its position on the target's structure, generating a 3D-to-2D keypoint catalog to be used with IP algorithms compatible with camera-based navigation. Additionally, the target's limb (or contour) in each keyframe is locally sampled into control points using edge detection. The edge points are converted to 3D using the depth map and grouped into 3D straight keylines. It was found that existing keyline descriptors were not mature enough for the present application, so alternative strategies were instead designed (Section VI). The offline training stage is illustrated in red in Fig. 2.

The online stage has the purpose of providing a fine pose estimate based on local feature matching after the closest keyframe has been found using coarse pose classification. If no estimate of the pose, $\hat{u} \in \mathcal{U} \cong SE(3)$, has been determined, local features are matched by detection: keypoints from the database pertaining to the current keyframe are matched by brute-force to the ones detected in the camera image, whereas the edges are matched by aligning the keyframe contour to the camera image contour in the least squares sense. Otherwise, the features are matched by tracking. This is not meant in the typical sense that the features are propagated from one camera image to the next, but instead the search space is reduced by reprojecting them from 3D into 2D based on $\hat{u}$ (Section VI.B).

The feature matches are processed separately and used to generate direct pseudo-measurements of the 6 degrees of freedom (DOF) relative pose. This is achieved by minimizing the reprojection error using Levenberg-Marquardt (LM) (Section VI.A) in an M-estimation framework (Section VI.D), which implements the rejection of outlying matches. The measurements are fused with an EKF to produce an estimate of the relative pose and velocity (Section VII). Both the M-estimator and the filter are accordant in representing the pose error as an element of $\mathfrak{se}(3)$, meaning that the measurement covariance determined from the former is used directly as the measurement noise in the latter, avoiding the need for tuning. The pose predicted by the filter for the following time-step is used to select the next keyframe and in the matching by tracking, providing temporal consistency. The online stage is summarized in blue in Fig. 2.

V. Coarse Pose Classification

A. Viewsphere Sampling

The concept of this module is to recover the viewpoint of the three-dimensional target object imaged in a two-dimensional scene using its pre-computed and known CAD model. The objective is to provide an initial, coarse, estimate of the appearance of the object based on its view classification so that then more precise pose estimation algorithms can be used to refine its pose.

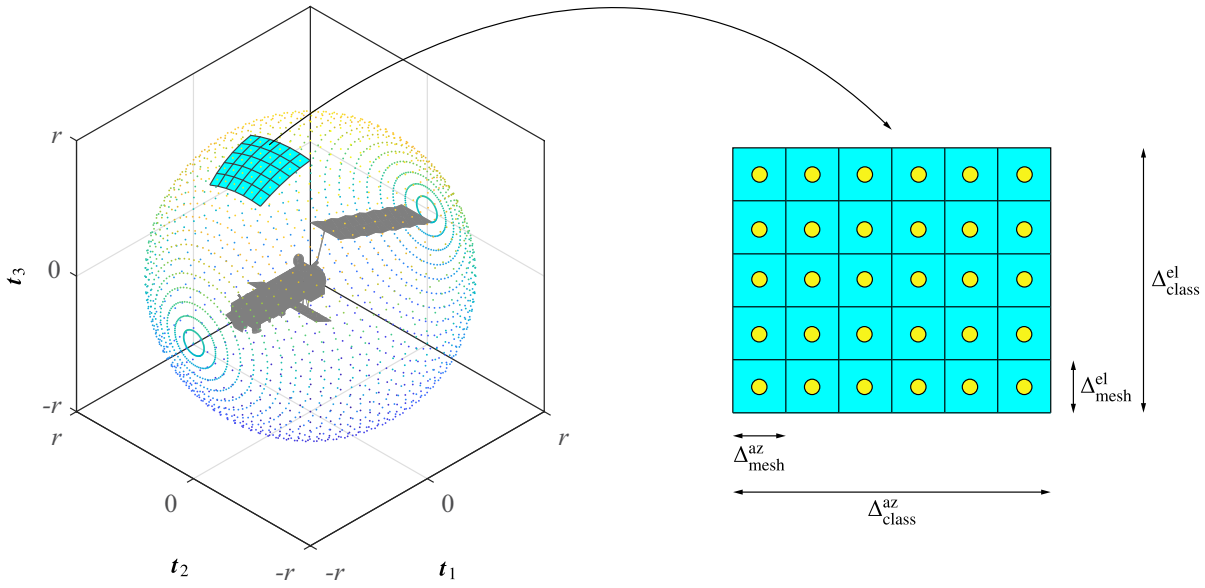


Fig. 3 The viewsphere for aspect sampling of a spacecraft (not to scale).

In order to capture the full three-dimensional aspect of the target, sampled views from the CAD are generated by resorting to the concept of the viewsphere: the model is located at the centre of a sphere, on the surface of which several cameras are placed, pointed at its centre of mass. The necessary viewpoints can be obtained by varying the spherical coordinates of the camera's position, i.e. the azimuth, elevation, and distance. The viewsphere is illustrated in Fig. 3. Each dot represents a camera position on the target body frame $\mathcal{F}_\mathcal{T}$ that will sample a view. Regarding the training of the sampled data, two different approaches using this viewsphere can be outlined. The first approach involves treating each dot on the viewsphere as a class. This has the immediate disadvantage that if a very fine mesh is defined (low Δ_{mesh}), the classes will not be distinctive enough, which could affect the performance of the view classification. On the other hand, selecting a high Δ_{mesh} does not solve the issue that each class will have only exactly one training image to use for the classification scheme. In order to solve both problems, a second approach is adopted in which dots are grouped into patches of width Δ_{class} to form a class, illustrated as the cyan patch in Fig. 3.

B. Global Feature Description

The following step is to select a measure to mathematically describe each training image obtained as explained above. Such a descriptor will be the basis to establish a correspondence between two viewpoints. The choice for a descriptor for viewpoint classification was motivated by by two main points: i) it must be a global representation of the target and ii) it must be robust to changes likely to be experienced during a space imaging scenario. The first point is justified by the fact that the goal is a classification of the aspect of the target, i.e. what is the view from the database that most closely resembles what the camera is observing. While it is possible in theory to use local descriptors for this task, when considering a spacecraft as the target, the same local features can be expected to be present in multiple views (e.g.

those sampled from multi-layer insulation (MLI) or solar panels), which would make the view classification harder. The second point refers to robustness against the model and what is actually observed during the mission; since modeling all the expected cases would be intractable, the descriptors should be resilient towards these, namely: translation, rotation, and scale changes (i.e. the expected 6-DOF in space), off-center perspective distortions, and illumination changes.

1. Image Moments

One type of descriptor that satisfies the above requirements are image moments. Moments are projections of an image $\mathfrak{I}$ onto a d-variable polynomial basis $\chi_{\boldsymbol{n}}$, with $\boldsymbol{n} = (n_1, \ldots, n_d)$ of the space of image functions defined on Π [65]. Formally:

$$M_{\boldsymbol{n}} = \int_{\Pi} \chi_{\boldsymbol{n}}(z)\mathfrak{I}(z)\,\mathrm{d}z, \tag{14}$$

$\mathfrak{I}(z)$ denotes the intensity value of pixel z in the image. Taking $\chi_{\boldsymbol{n}}(z) = z^{\boldsymbol{n}}$ leads to the well known geometric image moments that describe the "mass distribution" of the image: M_{00} is the mass of the image, M_{10}/M_{00} and M_{01}/M_{00} define the centroid, and so on. Moment computation over a regular image is dependant on the intensity value $\mathfrak{I}(z)$ of each pixel. This implies that the result of Eq. (14) will not be robust to illumination changes. Normalizing the image would provide global illumination invariance, but not local, therefore another strategy is needed. To this end, the viewpoint image is first binarized before computing the moments. This involves processing the image such that the resulting pixel intensities are mapped to either $\mathfrak{I}(z) = 0$ or $\mathfrak{I}(z) = 1$. In this way, the target is analyzed in terms of its shape, independently of how each patch is illuminated.

2. Complex Zernike Moments

Consider the Zernike moment (ZM) of the n-th degree with repetition ℓ, defined in 2D polar coordinates as [65]:

$$A_{n\ell} = \frac{n+1}{\pi} \int_0^{2\pi} \int_0^1 V^*_{n\ell}(r,\theta) f(r,\theta) r\,\mathrm{d}r\mathrm{d}\theta, \tag{15}$$

where

$$V^*_{n\ell}(r,\theta) = R_{n\ell}(r)e^{i\ell\theta} \quad \text{with} \quad n = 0, 1, 2, \ldots \quad \text{and} \quad \ell = -n, -n+2, \ldots, n$$
$$R_{n\ell}(r) = \sum_{s=0} (n - |\ell|)/2 (-1)^s \frac{(n-s)!}{s!\left(\frac{n+|\ell|}{2} - s\right)!\left(\frac{n-|\ell|}{2} - s\right)!} r^{n-2s}.$$

and $(\cdot)^*$ denotes complex conjugation. ZMs have two main attractive properties. Firstly, they are circular moments, meaning they change under rotation in a simple way which allows for a consistent rotation invariant design. Secondly,

they are orthogonal moments, which means that they present significant computational advantages with respect to standard moments, such as low noise and uncorrelation. Additionally, orthogonal moments can be evaluated using recurrent relations.

Since they carry these two traits, ZMs are said to be orthogonal on a disk. Hence, in order to compute the moments, the image must be appropriately pre-processed so that it is fully contained in one. By taking this disk to be the unit disk, scale invariance is achieved. Scale invariance is obtained when the image is mapped to the unit disk before calculation of the moments. Translation invariance is obtained by changing the coordinate system to be centered on the centroid. Regarding rotation invariance, one option occasionally seen is to take the ZM as the magnitude $|A_{n\ell}|$. This is not a recommended approach, as essentially the descriptor is cut in half, leading to a likely loss in recognition power. Instead, this work will deal explicitly with both real and complex parts of each ZM, in which case rotation invariance can be achieved by normalizing with an appropriate, non-zero moment $A_{m'\ell'}$ (typically A_{31}).

A fast computation of the Zernike polynomials up to a desired order can be obtained recursively since any set of orthogonal polynomials obeys a recurrent relation for three terms; in the case of ZMs the following formula has been developed by Kintner [66]:

$$k_1 R_{n+2,\ell}(r) = (k_2 r^2 + k_3) R_{n\ell}(r) + k_4 R_{n-2,\ell}(r), \tag{16}$$

where k_i, $i = 1, \ldots, 4$ are constants dependant on n and ℓ.

C. Training the Data

Given the process of generating the data and its descriptors, the final step is defining the classification method. The classifier algorithm shall recognize the aspect of the target given a database of ZM descriptor representation of viewpoints. Given the large volume of data involved, a Bayesian classifier is considered for this task, where the probability density function of each class is approximated using Gaussian mixture models (GMMs).

1. Bayesian Classification

Given a specific class $\mathcal{C}_m$, $m = 1, \ldots, k$, and a d-dimensional feature vector $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$, a Bayesian classifier works by considering $\boldsymbol{y}$ as the realization of a random variable $\boldsymbol{Y}$ and maximizing the posterior probability $P(\mathcal{C}_m|\boldsymbol{y})$, i.e. the probability that the feature vector $\boldsymbol{y}$ belongs to $\mathcal{C}_m$. This probability can be estimated using Bayes' formula [67]:

$$P(\mathcal{C}_m|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\mathcal{C}_m)\, P(\mathcal{C}_m)}{\sum_{i=1}^{k} p(\boldsymbol{y}|\mathcal{C}_i) P(\mathcal{C}_i)}. \tag{17}$$

The denominator is independent from $\mathcal{C}_m$ and hence can be simply interpreted as a scaling factor ensuring $P(\mathcal{C}_m|\boldsymbol{y}) \in [0, 1]$. Therefore, maximizing the posterior is equivalent to maximizing the numerator in Eq. (17).

The prior probability, $P(\mathcal{C}_m)$, expresses the relative frequency with which $\mathcal{C}_m$ will appear during the mission scenario; for a general case where one has no prior knowledge of the relative motion, an equiprobable guess can be made and the term can be set to $1/N$ for any m. The challenge is therefore to estimate the likelihood $p(\boldsymbol{y}|\mathcal{C}_m)$ of class $\mathcal{C}_m$, which is given by the respective probability density.

2. Gaussian Mixture Modeling

The Gaussian distribution is frequently used to model the probability density of some dataset. In the scope of the present work, it may prove overly optimistic to assume that all elements of the ZM descriptor vectors for each class are clustered into a single group. On the other hand, it can be too restrictive to model their distribution using hard-clustering techniques in case boundaries are not well defined. A more controllable approach to approximate a probability density function, while keeping the tractability of a normal distribution, is to assume the data can be modelled by a mixture of Gaussians:

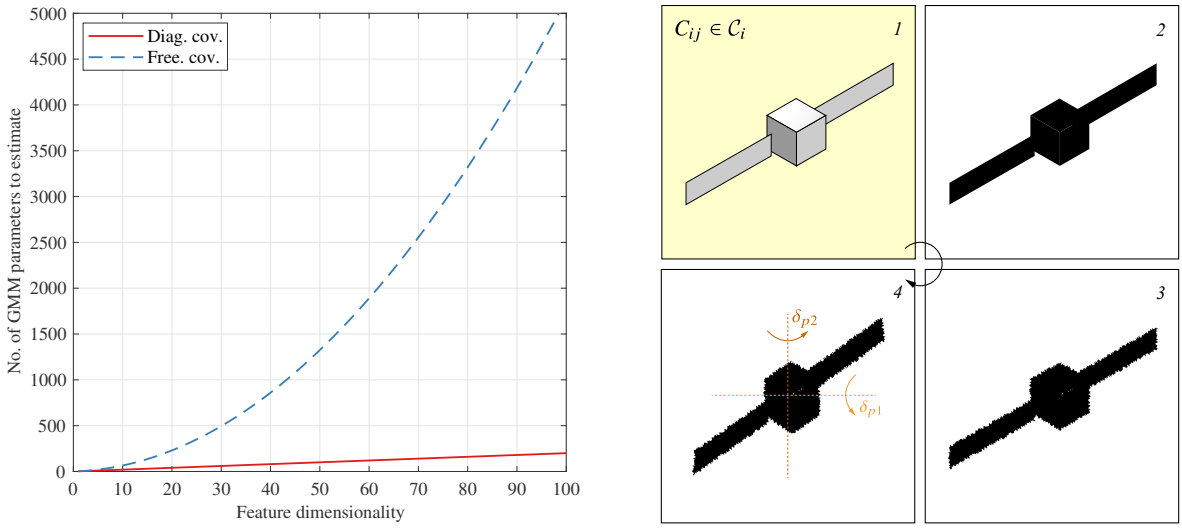
$$p(\boldsymbol{y}|\boldsymbol{\theta}) = \sum_{i=1}^{n} \alpha_i \mathcal{N}\left(\boldsymbol{y}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right), \quad \text{with} \quad \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{\mu})\right), \tag{18}$$

where α_i are scalar weighing factors, n is the number of mixture components, $\boldsymbol{\mu}$ denotes the mean vector, and $\boldsymbol{\Sigma}$ the covariance matrix, and $\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \alpha_1, \ldots, \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n, \alpha_n\}$ is the full set of parameters required to define the GMM.

When the number of mixture components n is known, the "optimal" mixture for each class, in the maximum likelihood (ML) estimation sense, can be determined using the classical expectation-maximization (EM) algorithm. EM works on the interpretation that the set of known points $\mathcal{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m\}$ is part of a broader, complete, data set $\mathcal{X} = \mathcal{Y} \cup \check{\mathcal{Y}}$ that includes unknown features [67]. In the case of GMMs, or finite mixtures in general, $\check{\mathcal{Y}} = \{\check{\boldsymbol{y}}_1, \ldots, \check{\boldsymbol{y}}_m\}$ can be defined as the set of m labels denoting which component generated each sample in $\mathcal{X}$. Each $\check{\boldsymbol{y}}_i = (\check{y}_{i,1}, \ldots, \check{y}_{i,n})^\top$ is a binary vector such that $\check{y}_{i,p} = 1, \check{y}_{i,q} = 0$ for all $p \neq q$ if sample $\check{\boldsymbol{y}}_i$ has been produced by component p.

However, the number of components is usually not known a priori. There are several methods to iteratively estimate the n; for this work the method of Figueiredo and Jain [68] is adopted. The algorithm provides an alternative to the generation of several candidate models, with different numbers of mixture components, and subsequent selection of the best fit, as this approach would still suffer from the drawbacks of EM; namely, the fact that it is highly dependant on initialization, and the possibility of one of the mixtures' weight α_i approaching zero (i.e. the boundary of the parameter space) and the corresponding covariance becoming close to singular. Instead, Figueiredo and Jain's method aims to find the best overall model directly. This is achieved by applying the minimum message length criterion to derive the following cost function for finite mixtures:

$$L(\boldsymbol{\theta}, \mathcal{Y}) = \frac{N}{2} \sum_{i=1}^{n} \ln\left(\frac{m\alpha_i}{12}\right) + \frac{n}{2} \ln\frac{m}{12} + \frac{n(N+1)}{2} - \ln p(\mathcal{Y}; \boldsymbol{\theta}), \tag{19}$$



(a) Necessary mixture parameters *vs.* d for $n = 1$ **(b) Artificial transformations on training data**

Fig. 4 Creating a training population.

where N is the number of parameters specifying each component. A modified EM is utilized to minimize Eq. (19), with the M-step given by:

$$\hat{\alpha}_p(k+1) = \frac{\max\left\{0, \left(\sum_{i=1}^{m} w_{i,p}\right) - \frac{N}{2}\right\}}{\sum_{j=1}^{n} \max\left\{0, \left(\sum_{i=1}^{m} w_{i,j}\right) - \frac{N}{2}\right\}}, \quad \hat{\boldsymbol{\theta}}_m(k+1) \leftarrow \arg\max_{\boldsymbol{\theta}_m} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(k)), \tag{20}$$

for $m = 1, \ldots, n$, where $t = k, k+1$ are sequential time-steps, and $w_{i,p} \in \mathcal{W} \coloneqq E[\check{\mathcal{Y}}|\mathcal{Y}; \hat{\boldsymbol{\theta}}(k)]$ is the a posteriori probability that $\check{y}_{ip} = 1$ after observing $\boldsymbol{y}_i$, computed as in the regular EM. The modified M-step performs explicit component annihilation, meaning that when one of the m components becomes unsupported by the data (i.e. close to zero), it is removed, thus impeding the algorithm from approaching the boundary of the parameter space. On the other hand, robustness towards initialization is achieved by starting the procedure with a large n and iteratively removing the unnecessary ones. If n is too large, it may occur that no component is granted enough initial support, leading the $\hat{\alpha}_i$ to be underdetermined. This is avoided by performing a component-wise update, i.e. recomputing $\mathcal{W}$ every time each element $\alpha_i, \boldsymbol{\theta}_i$ is updated, rather than doing it until the last $i = n$; in this way, if one component dies off, its probability mass is automatically redistributed to the other components, increasing their chance of survival. The proposed modifications will allow the modeling of each training class as a probability density in an unsupervised way.

3. Remarks

This section is concluded with some practical observations on the training procedure. The number of free parameters on a GMM will depend on the dimensionality of the data d, on the number of mixture parameters n, and on the constraints placed on the covariance $\boldsymbol{\Sigma}$. A "free" covariance matrix will have $1/2(d^2 + d)$ independent elements, since

it is symmetric, and hence the total number of mixture parameters will be $(1/2d^2 + 3/2d + 1)n - 1$. On the other hand, the covariance can be assumed as diagonal, in which case the total number of parameters to estimate becomes $2nd - 1$. Figure 4a plots the evolution of the number of parameters to estimate for a free covariance matrix and for a diagonal one in terms of the dimensionality of the features for $n = 1$. It can be considered as a lower bound for the number of samples m to be used in the training. The quadratic term in the free covariance case quickly diminishes the tractability of the problem when d is increased, which can pose a problem when training data is limited.

In [69], it was shown that the recognition power of complex ZMs for image retrieval begins to plateau beyond moments of the tenth order, which corresponds to approximately $d = 60$. This corresponds to 1890 parameters to be estimated for the free covariance case, while only 120 are necessary if a diagonal covariance is assumed. Since the ZMs are orthogonal, the correlation between moments is minimized and a diagonal covariance is an acceptable approximation. However, even if adjacent keyframes are grouped to form classes, the generated data might not be enough in terms of training. To this end, each keyframe C_{ij} imaged for class $\mathcal{C}_i$ is subjected to an image augmentation pipeline before the ZMs are computed and added to the training pool (Fig. 4b). This involves adding perturbations to the close contour (limb) of the binarized target shape and adding small perspective distortions to the image. Data augmentation is further explored in Section VIII.

VI. Motion Estimation

A. Problem Statement

The problem of solving the 2D-3D point correspondences for the 6-DOF pose of a calibrated camera is termed Perspective-n-Problem (PnP) and has a well-known closed form solution for $n = 3$ points (P3P). It relies on the fact that the angle between any image plane points z_i, z_j must be the same as the angle defined between their corresponding world points $\boldsymbol{p}_i, \boldsymbol{p}_j$ [56]. Additional methods have been developed for $n \geq 4$, such as EPnP [70], which expresses the n 3D points as a weighted sum of four virtual control points and then estimates the coordinates of these control points in the camera frame $\mathcal{F}_C$. While relatively fast to compute, these methods are notwithstanding less robust to noise and fail in the presence of erroneous correspondences. On the other hand, iterative approaches that take these aspects into account, giving the best possible estimate of the pose under certain assumptions are often called the "gold standard" algorithm [71]. In this section, given an initial, coarse evaluation of the relative pose, an iterative refinement of its estimate based on nonlinear manifold parameterization is proposed.

Let u represent the object to be determined. Its domain is a manifold $\mathcal{U} \subset \mathbb{R}^m$ which defines the parameter space. Let $\boldsymbol{a}$ be the vector of measurements in $\mathbb{R}^n$. Suppose $\boldsymbol{a}$ is observed in the presence of noise with a covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{a}}$, and let $\bar{\boldsymbol{a}}$ be its true value, i.e. $\boldsymbol{a} = \bar{\boldsymbol{a}} + \Delta\boldsymbol{a}$. Let $\boldsymbol{h} \colon \mathbb{R}^m \to \mathbb{R}^n$ be a mapping such that, in the absence of noise, $\boldsymbol{h}(\bar{u}) = \bar{\boldsymbol{a}}$. Varying the value of $\bar{u}$ traces out a manifold $\mathcal{A} \subset \mathbb{R}^n$ defining the set of allowable measurements, i.e. the measurement

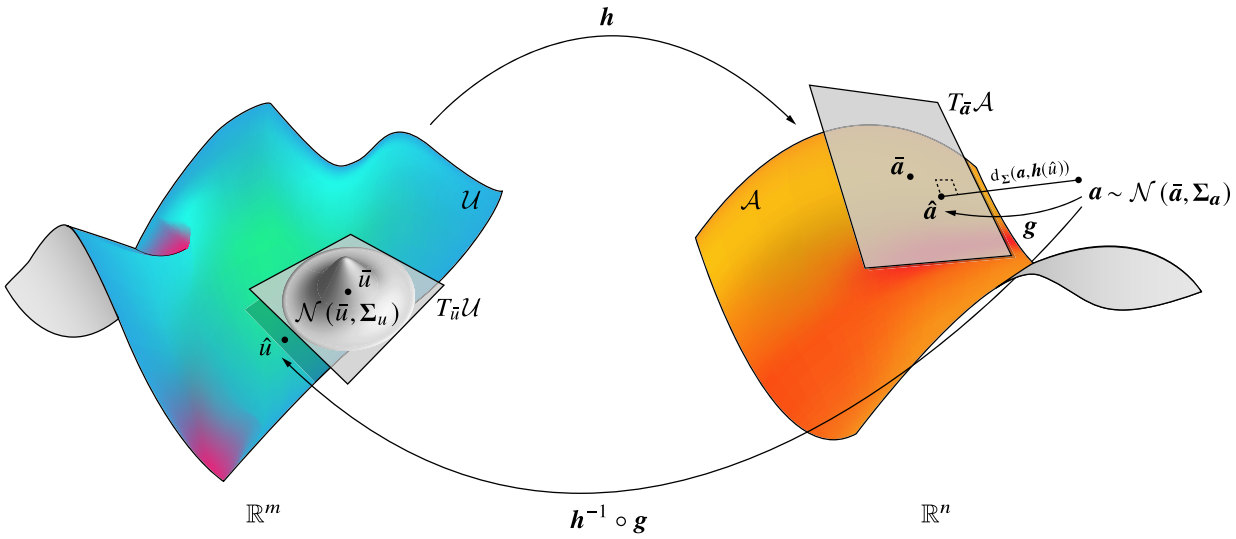


Fig. 5 Geometric correction and parametric fitting on manifolds.

space. The objective is, given a measurement $\boldsymbol{a}$, to find the vector $\hat{\boldsymbol{a}} \in \mathbb{R}^n$ lying on $\mathcal{A}$ that is closest to $\bar{\boldsymbol{a}}$ (Fig. 5).

Given the form of the multivariate normal probability density function, under the assumption of Gaussian noise, it is straightforward that the maximum likelihood (ML) solution is obtained by minimizing the Mahalanobis distance:

$$\mathrm{d}_{\boldsymbol{\Sigma}_{\boldsymbol{a}}}\left(\boldsymbol{a}, \boldsymbol{h}(\hat{u})\right) \coloneqq \left(\left(\boldsymbol{a} - \boldsymbol{h}(\hat{u})\right)^{\top} \boldsymbol{\Sigma}_{\boldsymbol{a}}^{-1} \left(\boldsymbol{a} - \boldsymbol{h}(\hat{u})\right)\right)^{1/2}, \quad \text{with } \boldsymbol{h}(\hat{u}) = \hat{\boldsymbol{a}}. \tag{21}$$

It is reasonable in most cases to assume that, in the neighborhood of $\bar{\boldsymbol{a}}$, the surface of $\mathcal{A}$ is essentially planar and well approximated by the tangent space within the order of magnitude of the measurement noise variance [71]. Then, the ML corrected measurement $\hat{\boldsymbol{a}}$ is the foot of the perpendicular from $\boldsymbol{a}$ onto the tangent plane. The benefit of this approximation is that it allows the measurement residual error to be modelled as a Gaussian distribution in the normal space of $\mathcal{A}$, whereas the measurement estimation error is a Gaussian distribution in the tangent space $T_{\bar{\boldsymbol{a}}}\mathcal{A}$. In computer vision it is common to have image or world points as measured variables, so typically one can safely write the estimation error as $\hat{\boldsymbol{a}} - \bar{\boldsymbol{a}}$. Analogously, the parameter estimation error $\delta\boldsymbol{u}$ is, to a first approximation, constrained to be in the tangent space $T_{\bar{u}}\mathcal{U}$. In general terms, the present work shall assume this local distribution approximation is valid for small errors whenever dealing with the probability distribution of a variable constrained to a manifold [72]. Let $\boldsymbol{g} \colon \mathbb{R}^n \to \mathcal{A}$ map a point to the surface of the measurement space, as defined in Eq. (21). Assuming that $\boldsymbol{h}$ is invertible such that $\boldsymbol{h}^{-1} \colon \mathcal{A} \to \mathbb{R}^m$, then the mapping $\boldsymbol{h}^{-1} \circ \boldsymbol{g}$ can be used to propagate the measurement noise covariance $\boldsymbol{\Sigma}_{\boldsymbol{a}}$ to obtain the covariance matrix of the ML estimate $\bar{u}$.

Let $\varphi(u) = \mathrm{d}^2_{\boldsymbol{\Sigma}_{\boldsymbol{a}}}\left(\boldsymbol{a}, \boldsymbol{h}(u)\right)$ for succinctness. Then, the problem can be posed as:

$$\hat{u} = \underset{u \in \mathcal{U}}{\arg\min}\, \varphi(u). \tag{22}$$

When the function $\boldsymbol{h}$ is nonlinear, Eq. (22) may be solved iteratively by linearizing it around a reference parameter $\varphi(u_0)$. In the case where the parameter space $\mathcal{U}$ can be identified with Euclidean space, linearizing and differentiating $\varphi(\boldsymbol{u})$ at $\boldsymbol{u}_0$, $\boldsymbol{u} \in \mathbb{R}^m$, leads to the well-known normal equations yielding a correction $\Delta\boldsymbol{u}$ at iteration $t = k$ such that $\hat{\boldsymbol{u}}_{k+1} = \hat{\boldsymbol{u}}_k + \Delta\boldsymbol{u}$. If not, this update is not valid: as noted in Section III, $\boldsymbol{u}_{k+1}$ is not guaranteed to be an element of $\mathcal{U}$.

One possible solution is to nevertheless apply the correction via standard addition and then project the result back to the parameter manifold $\mathcal{U}$, which could introduce additional noise in the system and drive the result away from the ML estimate $\hat{u}$. A more elegant alternative solution is to exploit the local Euclidean structure of $\mathcal{U}$ around u_0 to generate a new set of normal equations. Taking $\mathcal{U} \cong SE(3)$ and using the composition operator from Eq. (13), linearizing $\varphi(u)$ yields:

$$\varphi(u) \approx \varphi(u_0) + (u \ominus u_0)^\top \nabla\varphi\big|_{u \ominus u_0 = 0} . \tag{23}$$

Equation (23) thus motivates working with the pose estimation error $\delta\boldsymbol{u} = u \ominus u_0$ explicitly, which is an element of $\mathfrak{se}(3)$. This can be shown to lead to the normal equations of the form:

$$\boldsymbol{J}^\top \boldsymbol{\Sigma}_{\boldsymbol{a}} \boldsymbol{J} \delta\boldsymbol{u} = -\boldsymbol{J}^\top \boldsymbol{\Sigma}_{\boldsymbol{a}} \boldsymbol{r}, \tag{24}$$

where $\boldsymbol{r} = \boldsymbol{h}(u_0) - \boldsymbol{a}$ is the residual vector. The other advantage of Eq. (24) is that the Jacobian matrix $\boldsymbol{J} \coloneqq \frac{\partial h(u_0 \oplus \delta\boldsymbol{u})}{\partial(\delta\boldsymbol{u})}|_{\delta\boldsymbol{u}=\boldsymbol{0}}$ is computed with respect to the basis of $\mathfrak{se}(3)$. At the end of each iteration, the updated parameter is obtained via the exponential map by following Eq. (13), thus ensuring it naturally remains an element of $\mathcal{U}$.

B. Structural Model Constraints

1. From Visual Point Feature Correspondences

This subsection explains how to find a relationship between some pre-existing knowledge of the target's structure and measurements taken of it with a digital camera that allows for the relative pose to be estimated in accordance with the theory developed above. Note that Eq. (1) describes such a relationship, as $\boldsymbol{z}$ is the reprojection in the image plane of a 3D point $\boldsymbol{p}$ defined in $\mathcal{F}_{\mathcal{T}}$. Therefore, given a number of m correspondences $\boldsymbol{z}_i \leftrightarrow \boldsymbol{p}_i$ between 2D image points and 3D structural points, the task is to find $\boldsymbol{T}$ such that $\boldsymbol{z}_i = \boldsymbol{\pi}(\boldsymbol{K}\boldsymbol{T} \otimes \boldsymbol{p}_i)$, for all $i = 1, \ldots, m$. Of course, the relative pose does not have to be represented in the homogeneous form $\boldsymbol{T}$; the problem is simply posited as such because of the significance of Eq. (1) in the computer literature and, as shall be seen, because it leads to a simple form of the Jacobian.

The obvious difficulty in this formulation lies in solving the feature correspondence problem due to the topological difference between $\boldsymbol{z}_i$ and $\boldsymbol{p}_i$. The proposed work solves this difference by attributing to each structural point a representation on the image plane in an offline training stage. This is achieved as follows: in each model keyframe, point features (keypoints) are selected by a detector algorithm as centers of regions of interest (blobs) in the image.

These regions are typically deemed "interesting" when they have a defining property (e.g. brightness) that differs from their surroundings. Once detected, a keypoint can then be extracted by applying a descriptor algorithm which will encode its characterizing traits into a vector. This descriptor vector normally incorporates information about the blob as well, allowing keypoints to be matched in different images of the same object in a robust manner with respect to changes in scale, orientation, and brightness, among others. Since the relative pose is known for each keyframe, Eq. (1) is inverted to generate a ray passing through each keypoint. The depth of the ray is the image of the keypoint in the depth map corresponding to that keyframe, thus determining the equivalent 3D structural point. In this way, each $\boldsymbol{p}_i$ is annotated offline with a 2D descriptor computed from the reprojection $\boldsymbol{z}'_i$ in the current keyframe. Then, computing a descriptor vector for the keypoints detected online $\boldsymbol{z}_i$ grants the equivalence $\{\boldsymbol{z}_i \leftrightarrow \boldsymbol{p}_i\} \Leftrightarrow \{\boldsymbol{z}_i \leftrightarrow \boldsymbol{z}'_i\}$, reducing a 3D-2D correspondence problem to a 2D-2D one.

Since the structural points $\boldsymbol{p}_i$ are obtained via ground truth depth maps for keyframes with perfectly known $\boldsymbol{T}$, they are considered to be measured with maximum accuracy, and the error is thus concentrated in the measured image points $\boldsymbol{z}_i$. In other words, the measurement space $\mathcal{A}$ is a manifold embedded in $\mathbb{R}^{2m}$ (i.e. $\boldsymbol{a}$ is construed by stacking the x and y components of all $\boldsymbol{z}_i$) and the parameter space $\mathcal{U}$ is 6-dimensional (i.e. the dimensions of $\mathfrak{se}(3)$). Furthermore, as each measured image point is obtained algorithmically with the same feature detector, each $\boldsymbol{z}_i$ is modelled as a random variable sampled from an isotropic (Gaussian) distribution. The ML estimate of the pose is in this manner obtained by minimizing the geometric error (cf. Eq. (21)) which is reduced to the standard squared Euclidean distance:

$$\hat{\boldsymbol{T}} = \underset{\boldsymbol{T} \in SE(3)}{\arg\min} \sum_{i=1}^{m} \mathrm{d}\left(\boldsymbol{z}_i, \boldsymbol{\pi}\left(\boldsymbol{K}\boldsymbol{T} \otimes \boldsymbol{p}_i\right)\right)^2 . \tag{25a}$$

The solution to Eq. (25a) is found iteratively via LM with the Jacobian matrix given by:

$$\begin{aligned} \boldsymbol{J}_p^s = \left.\frac{\partial \boldsymbol{\pi}(\boldsymbol{K}(\boldsymbol{T} \oplus \boldsymbol{\varepsilon}) \otimes \boldsymbol{p})}{\partial \boldsymbol{\varepsilon}}\right|_{\boldsymbol{\varepsilon}=\boldsymbol{0}} &= \left.\frac{\partial \boldsymbol{\pi}'(\boldsymbol{p}')}{\partial \boldsymbol{p}'}\right|_{\substack{\boldsymbol{\pi}'(\boldsymbol{p}') := \boldsymbol{\pi}(\boldsymbol{K}\boldsymbol{p}') \\ \boldsymbol{p}' = \boldsymbol{T} \otimes \boldsymbol{p}}} \left.\frac{\partial \boldsymbol{T}' \otimes \boldsymbol{p}}{\partial \boldsymbol{T}'}\right|_{\boldsymbol{T}' = \boldsymbol{T} \oplus \boldsymbol{\varepsilon} = \boldsymbol{T}} \left.\frac{\partial \exp(\boldsymbol{\varepsilon})\boldsymbol{T}}{\partial \boldsymbol{\varepsilon}}\right|_{\boldsymbol{\varepsilon}=\boldsymbol{0}} \\ &= \begin{bmatrix} \frac{f_x}{p'_3} & 0 & -f_x \frac{p'_1}{p'^2_3} & -f_x \frac{p'_1 p'_2}{p'^2_3} & f_x\left(1 + \frac{p'^2_1}{p'^2_3}\right) & -f_x \frac{p'_2}{p'_3} \\ 0 & \frac{f_y}{p'_3} & -f_y \frac{p'_2}{p'^2_3} & -f_y\left(1 + \frac{p'^2_2}{p'^2_3}\right) & f_y \frac{p'_1 p'_2}{p'^2_3} & f_y \frac{p'_1}{p'_3} \end{bmatrix}, \end{aligned} \tag{25b}$$

where f_x, f_y is the focal length and $\boldsymbol{\varepsilon} \in \mathfrak{se}(3)$ is a small perturbation of the pose.

Note that the minimization scheme is built on the assumption that the structural points are known far more accurately than the image points detected online [71]. This is a valid assumption for the context of the present study: since the CAD model of the target is given, accurate depth maps are produced to register 3D information on the training keyframes with virtually no error (Section IV). The actual source of error contaminating the minimization of the geometric error consists of possible outlying matches during the online stage, which are mitigated by the M-estimation module (Section

VI.D). This is also the case for structural edgs (see next subsection).

2. From Visual Edge Feature Correspondences

The structural model constraints may also be formulated in terms of different types of features, such as straight line segments. This is likewise an important element to consider in space relative navigation, as spacecraft often resemble cuboid shapes or are composed of elements shaped as such; therefore it is expected to have detectable line features (keylines) when imaging this kind of targets. It has been shown in this context that, while keypoints are more distinctive in the context of minimizing Eq. (25a), keylines are actually more robust in terms of preventing the solution from diverging [38].

In two dimensions, a point $\boldsymbol{z}$ lies on a line $\boldsymbol{l} = (l_1, l_2, l_3)^\top$ if $\boldsymbol{z}^\top \boldsymbol{l} = \boldsymbol{0}$. Assume there exist m correspondences $\boldsymbol{l}_i \leftrightarrow \boldsymbol{\ell}_i$ between the 2D lines and 3D lines. Therefore, one can formulate a geometric distance for 2D-3D line correspondences in terms of the reprojection of a point $\boldsymbol{p}_{ij} \in \boldsymbol{\ell}_i$ onto the image plane:

$$\hat{\boldsymbol{T}} = \underset{\boldsymbol{T} \in SE(3)}{\arg\min} \sum_{i=1}^{m} \sum_{j=0}^{k} \boldsymbol{l}_i^\top \boldsymbol{K}\boldsymbol{T} \otimes \boldsymbol{p}_{ij}, \tag{26}$$

The Jacobian matrix $\boldsymbol{J}_l^s$ corresponding to the minimization of Eq. (26) can be derived in a similar manner to Eq. (25b).

In practice, matching keylines is not as straightforward as matching keypoints, as the former are typically less distinctive than the latter. For the scope of this work, only the contour of the target is considered, which is discretized into a finite number of edge points that are assumed to belong to a (straight) keyline. Additionally, edge points can be registered in the same way as structural keypoints through the use of depth maps.

C. Local Feature Processing

1. Detection

Distinct point features are identified in an image of the target using the Oriented FAST and Rotated BRIEF (ORB) detector [13]. The basis of ORB is the Features from Accelerated Segment Test (FAST) algorithm [73], developed with the purpose of creating a high-speed keypoint finder for real-time applications[†]. It first selects a pixel $\boldsymbol{z}_i$ in the image as candidate. A circle of 16 pixels around $\boldsymbol{z}_i$ and a threshold α are defined. If there exists a set of n contiguous pixels in the circle which are all brighter than $I(\boldsymbol{z}_i) + \alpha$ or all darker than $I(\boldsymbol{z}_i) - \alpha$, then $\boldsymbol{z}_i$ is classified as a keypoint. The algorithm is made robust with an offline machine learning stage, training it to ignore regions in an image where it typically lacks interest points, thus improving detection speed. As the original method is not robust to changes in size or rotation, ORB applies a pyramidal representation of FAST for multi-scale feature detection and assigns an orientation to each one by defining a vector from its origin to the intensity baricenter of its support region. The choice for the

[†]Although ORB has also been developed for feature description, using a modification of the Binary Robust Invariant Scalable Keypoints (BRIEF) algorithm, it is applied herein for detection only.

keypoint algorithms (see also Section VI.C.2) is the product of the authors' previous survey on IP techniques for relative navigation in space [74].

The Canny algorithm is the basis for edge detection [75]. First, the image is filtered with a Gaussian kernel in order to remove noise. Secondly, the intensity gradients at each pixel are computed. Then, non-maximal suppression and a double thresholding are applied to discard spurious responses and identify edge candidates. The method from Ref. [76] is used to efficiently extract keylines from the edge image by incrementally connecting edge pixels in straight lines and merging those with small enough differences in overlap and orientation.

2. Description

For each detected keypoint, a binary string is generated encoding information about its support region using the Fast Retina Keypoint (FREAK) descriptor [77], which takes inspiration in the design of the human retina. The method adopts the retinal sampling grid as the sampling pattern for pixel intensity comparisons, i.e. a circular design with decreasing density from the center outwards, achieved using different kernel sizes for the Gaussian smoothing of every sample point in each receptive field; these overlap for added redundancy leading to increased discriminating power. Each bit in the descriptor thus represents the result of each comparison test, which only has two possible outcomes: either the first pixel is brighter than the second, or vice-versa. A coarse-to-fine pair selection is employed to maximize variance and uncorrelation between pairs. In this way, the first 16 bytes of the descriptor represent coarse information, which is applied as a triage in the matching process, and a cascade of comparisons is performed to accelerate the procedure even further.

3. Brute-Force Detection Matching

In an initial stage, the features are matched using brute force, since no estimate of the pose is yet available. In the case of the point features, this implies that all those detected in the initial frame are compared against those in the train keyframe. This is achieved by computing the Hamming distance $\mathrm{d}_{\mathrm{Ham}}(\cdot,\cdot)$ between their corresponding descriptors, i.e. the minimum number of substitutions required to convert one into the other. It can be swiftly computed by applying the exclusive-OR (XOR) operator followed by a bit count, which provides an advantage in terms of computational performance with respect to the Euclidean distance test used with more traditional floating point descriptors. For each query, the two closest train descriptor matches are selected and subjected to a nearest-neighbour distance ratio (NNDR) test: the matching of the descriptors $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ is accepted if

$$\frac{\mathrm{d}_{\mathrm{Ham}}(\boldsymbol{s}_i, \boldsymbol{s}_j)}{\mathrm{d}_{\mathrm{Ham}}(\boldsymbol{s}_i, \boldsymbol{s}_k)} < \mu_{\mathrm{NNDR}}, \tag{27}$$

where $\boldsymbol{s}_j, \boldsymbol{s}_k$ are the 1st and 2nd nearest neighbours to $\mathbf{s}_i$ and μ_{NNDR} is a ratio from 0 to 1.

As descriptors for edge features are not employed, an alternative strategy was devised to match them. The full contours $\mathcal{D}_q, \mathcal{D}_t \subset \mathbb{R}^2$ of the target in the query image and the train keyframe, respectively, are considered. Each contour is a set of n discretized edge points of the same size, i.e. $\mathcal{D}_q = \{\boldsymbol{z}_{q,1}, \ldots, \boldsymbol{z}_{q,n}\}$ and $\mathcal{D}_t = \{\boldsymbol{z}_{t,1}, \ldots, \boldsymbol{z}_{t,n}\}$. Even though the query image and train keyframe represent the same aspect of the target, there will be differences that are reflected on the contours. In particular, $\mathcal{D}_q$ and $\mathcal{D}_t$ will be different by a 2D affine transformation $\Lambda(\beta, \theta, \boldsymbol{t}) : \mathcal{D}_t \rightarrow \mathcal{D}_q$, where $\beta > 0$ is a scaling factor, $\theta \in [-\pi, \pi[$ rad is an angle of rotation and $\boldsymbol{t} = (t_1, t_2)^\top$ is a translation vector. The contour alignment problem is posed in the least squares sense as

$$\underset{\boldsymbol{t},\beta,\theta}{\arg\min}\, \mathrm{d}_{\mathrm{Fro}}\left(\mathcal{D}_q, \Lambda\left(\mathcal{D}_t\right)\right), \tag{28}$$

where $\mathrm{d}_{\mathrm{Fro}}(\cdot,\cdot)$ is the Frobenius distance. Because of the multiplicative trigonometric terms of Λ, Eq. (28) is nonlinear. However, the problem can be converted into an equivalent linear one by a change of variables [78]:

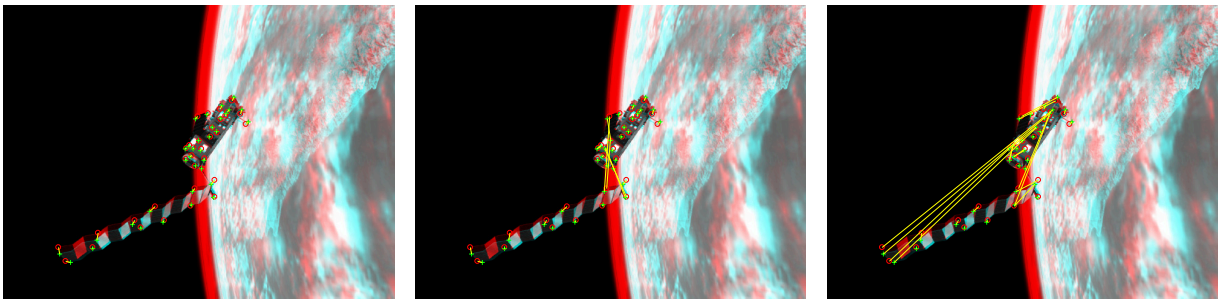
$$\underset{(t_1,t_2,b_1,b_2)\in\mathbb{R}^4}{\arg\min}\; \mathrm{d}\left(\begin{pmatrix} z_{q,1,1} \\ z_{q,1,2} \\ \vdots \\ z_{q,n,1} \\ z_{q,n,2} \end{pmatrix}, \begin{bmatrix} 1 & 0 & z_{t,1,1} & -z_{t,1,2} \\ 0 & 1 & z_{t,1,2} & z_{t,1,1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & z_{t,n,1} & -z_{t,n,2} \\ 0 & 1 & z_{t,n,2} & z_{t,n,1} \end{bmatrix} \begin{pmatrix} t_1 \\ t_2 \\ b_1 \\ b_2 \end{pmatrix}\right) \text{ with } \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \beta \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \Leftrightarrow \begin{pmatrix} \theta \\ \beta \end{pmatrix} = \begin{pmatrix} \arcsin\left(b_2 / \sqrt{b_1^2 + b_2^2}\right) \\ \sqrt{b_1^2 + b_2^2} \end{pmatrix}, \tag{29}$$

where $\boldsymbol{z}_{u,v} = (z_{u,v,1}, z_{u,v,2})^\top$. In this way, a global solution of the minimum can be calculated using standard linear algebra. However, Eq. (29) depends on the correspondences between the query and train edge points, which are not known a priori. To simultaneously solve for the edge point correspondence problem and contour alignment, the algorithm is modified by solving n linear least squares problems, each time shifting the order of the edge points in $\mathcal{D}^t$ by one, and selecting the minimum of the n residual norms. Thus, the only necessary inputs are two sets of sequential but not necessarily correspondent edge points.

4. Predictive Tracking Matching

Once the algorithm is initialized, knowledge of the current solution can be used to improve the performance of the feature matching processes. In particular, the predicted estimate of the pose output by the filtering module is used to help anticipate where the features will be located in the next frame in time, in this way introducing a temporal tracking constraint that improves the pose estimation accuracy.

In the case of point features, tracking matching is achieved by fitting a grid of $p \times q$ cells on the boundary of the target in the query camera image. The detected keypoins are binned into the resulting cells. Then, the 3D structural points of the currently selected database keyframe are reprojected onto the query image according to the predicted pose (cf. Eq. (1)) and equally binned according to the grid. Lastly, descriptor-based matching is applied on a per-cell basis,



(a) 0% outliers **(b) 25% outliers** **(c) 50% outliers**

Fig. 6 Sequential feature matching in false-color composite overlay.

vastly reducing the number of possible matching candidates. This step was found essential in order to maintain the accuracy of the algorithm during sequences where ambiguous modules are imaged (e.g. MLI) or when the query image is too distinct from the train one (e.g. due to reflections).

In the case of edge features, tracking matching is done by first detecting keylines on the query edge image. Each query keyline is then drawn on the image plane with a unique color. The 3D edge points and corresponding keylines from the train keyframe are reprojected onto the image plane. Then, the matching algorithm iterates over each reprojected edge point and a 1D search is performed perpendicularly to it according to the corresponding keyline, obtained in the offline training stage, until the closest colored pixel is found. Hence, 3D edge points are matched to 2D keylines satisfying the conditions to minimize Eq. (26).

D. Robust Estimation

When the measurements are assumed to have equal variance, the ML estimate is found by solving

$$\hat{u} = \underset{u \in \mathcal{U}}{\arg\min}\, \boldsymbol{r}^\top \boldsymbol{r}, \tag{30}$$

where $\boldsymbol{r} \coloneqq \mathrm{d}(\boldsymbol{a}, \boldsymbol{h}(u))$ is the residual vector. The problem becomes one of classical least squares (LS) estimation, where the covariance $\boldsymbol{\Sigma}_{\boldsymbol{a}}$ vanishes as Eq. (30) is equivariant with respect to scale. However, ordinary LS is not robust to outliers, i.e. spurious data that may contaminate the measurements. In the scope of this work, measurements are matches between features, which can be erroneous due to the typical space rendezvous scenario as imaged by a camera. For instance, a solar panel might resemble a repeating pattern that yields many features which look identical, or intense illumination from the Sun acting on the spacecraft can change its local aspect with respect to a model image. Consider Fig. 6, where sequential frames of a simulated rendezvous sequence are represented with a false-color overlay. Point feature matches are also represented and connected by lines for several levels of outlier contamination. Whereas, from the reader's perspective, it may seem that for a 25% outlier level (middle image) the true trajectory can still be determined, in theory the presence of a single outlier is enough to make the LS estimate diverge [79].

Robustness with respect to outliers can be achieved by generalizing Eq. (30) into an M-estimator:

$$\hat{u} = \underset{u \in \mathcal{U}}{\arg\min} \sum_{i=1}^{N} \rho\left(\frac{r_i}{\hat{\sigma}}\right), \tag{31}$$

where ρ is a symmetric, positive-definite function with subquadratic growth, and $\hat{\sigma}^2$ is an estimate of the variance, or scale, of $\boldsymbol{r}$. Solving Eq. (31) implies

$$\sum_{i=1}^{N} \psi\left(\frac{r_i}{\hat{\sigma}}\right) \frac{\mathrm{d}r_i}{\mathrm{d}u} \frac{1}{\hat{\sigma}} = 0, \tag{32}$$

where $\psi(x) \coloneqq \mathrm{d}\rho(x)/\mathrm{d}x$ is defined as the influence function of the M-estimator. This function measures the influence that a data point has on the estimation of the parameter u. A robust M-estimator $\rho(x)$ should meet two constraints: convexity in x, and a bounded influence function [80]. By acknowledging the latter point, it becomes clear why the general LS is not robust, since $\rho(x) = x^2/2$ and therefore $\psi(x) = x$.

There are two possible approaches to define the normal equations for M-estimation that avoid the computation of the Hessian [81]:

$$\boldsymbol{J}^\top \boldsymbol{J} \delta \boldsymbol{u} = -\boldsymbol{J}^\top \boldsymbol{\psi}\left(\frac{\boldsymbol{r}}{\hat{\sigma}}\right) \hat{\sigma}, \tag{33a}$$

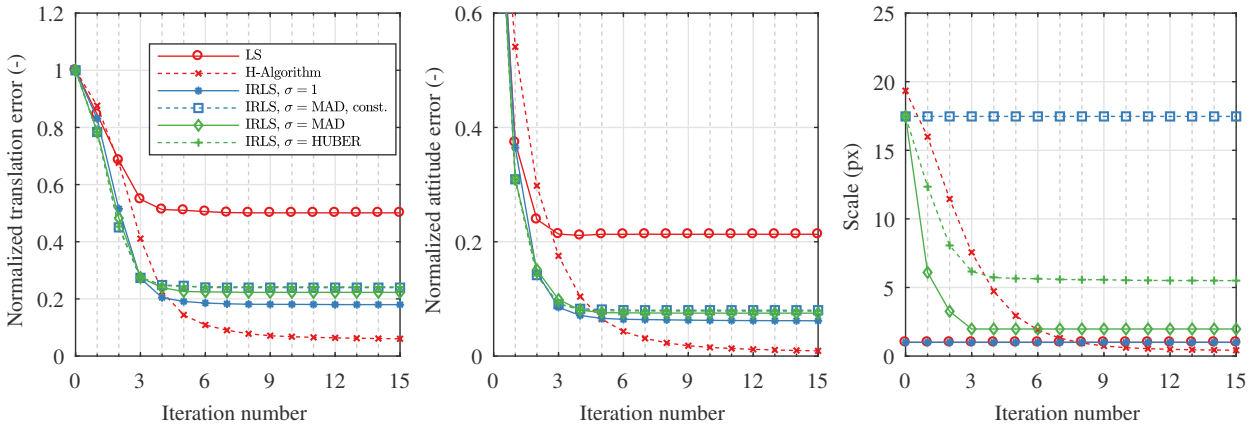
$$\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{J} \delta \boldsymbol{u} = -\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{r}, \tag{33b}$$

where $\boldsymbol{W} = \mathrm{diag}(w(r_1/\hat{\sigma}), \ldots, w(r_n/\hat{\sigma}))$ and $w(x) \coloneqq \psi(x)/x$. The first method was developed by Huber [82] and generalizes the normal equations through the modification of the residuals via ψ and $\hat{\sigma}$. Huber proposed a specific loss function, the Huber M-estimator $\rho_{\mathrm{Hub}}(x)$. Huber's algorithm provides a way to jointly estimate the scale σ alongside the parameter u with proven convergence properties. The minimization algorithm (e.g. LM) is simply appended with the procedure:

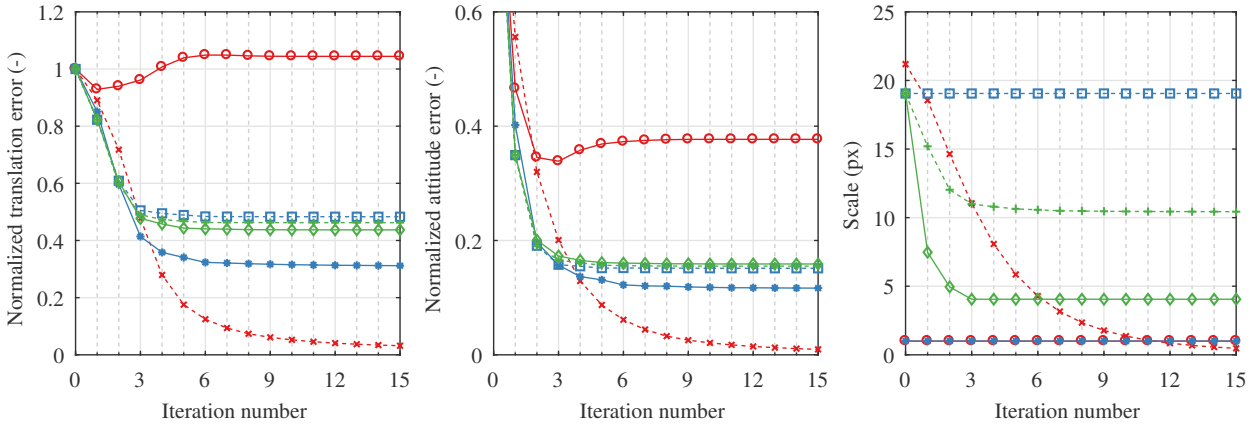
$$\sigma_{k+1}^2 = \frac{1}{(n-p)\beta} \sum_{i}^{n} \left(\frac{r_i}{\sigma_k}\right)^2 \sigma_k^2, \tag{34}$$

where β is a bias-correcting factor. The second method was developed by Beaton and Tukey [83] and is commonly known as iteratively reweighed least squares (IRLS), due to the inclusion of the weights matrix $\boldsymbol{W}$ that assumes the role of $\boldsymbol{\Sigma_a}$ (cf. Eq. (24)). Tukey proposed an alternative robust loss function, $\rho_{\mathrm{Tuk}}(x)$.

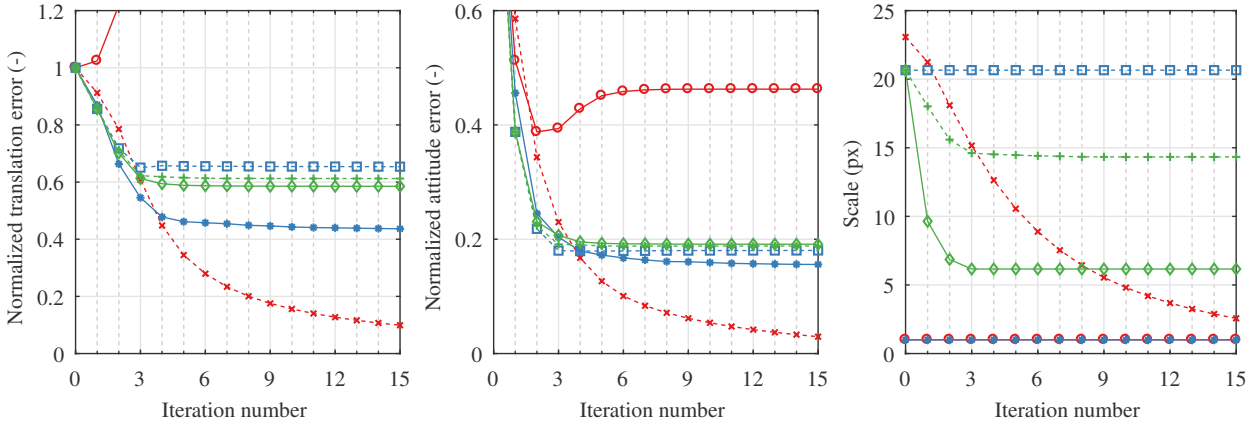
Each robust loss function, $\rho_{\mathrm{Hub}}(x)$ and $\rho_{\mathrm{Tuk}}(x)$, can be compared regardless of the formulation. The Huber M-estimator is considered to be adequate for almost all situations, but does not eliminate completely the influence of large errors [80]. On the other hand, the Tukey M-estimator is non-convex, but is a "hard redescender", meaning that its influence function tends to zero quickly so as to aggressively reject outliers, explaining its frequent use in computer vision applications, where the outliers typically have small residual magnitudes [84].



(a) 10% outliers



(b) 20% outliers



(c) 30% outliers

Fig. 7 Minimization of the reprojection function from the images of a randomly generated point cloud, averaged over 100 runs.

The scale estimation step warrants special attention. In several applications, it can be found that σ is often ignored and set to 1. This is erroneous since Eq. (31) is non-equivariant with respect to scale [79]. Whereas the Huber algorithm (Eq. (33a)) grants a procedure to jointly estimate the parameter and scale, convergence is not guaranteed when applying the scale estimation step to IRLS (Eq. (33b)) [85]. Instead, a common method when resorting to IRLS is to recursively estimate σ using the median absolute deviation (MAD) for the first few iterations, and then allowing the minimization to converge on u with fixed σ [80, 84].

In order to study the effect of scale estimation on the parameter estimation and to compare the different possible approaches, an experiment has been devised. First, a number of 3D world points is randomly sampled from the volume of a cube. They are subsequently projected onto the image plane according to a random pose. Points that fall outside the image plane are culled. Matches between 3D world points and 2D camera points are contaminated artificially with outliers. Then, the pose is M-estimated with $\rho_{\text{Hub}}(x)$ according to the cost function of Eq. (25a), where the initial guess is defined by contaminating the true pose with zero-mean, white, Gaussian noise. Five distinct methods are benchmarked: i) LS, ii) Huber's algorithm, iii) IRLS with $\sigma = 1$, iv) IRLS with σ estimated by one iteration of MAD, v) IRLS with σ estimated by three iterations of MAD, vi) IRLS with σ estimated by Huber's algorithm. The experiment is repeated for several trials.

The results are shown in Fig. 7. The pose estimation error is decomposed into translation and rotation normalized according to the initial guess. The evolution of the scale estimation is also shown. The percentage of outliers present in the data ranges from 10% to 30%. It can be seen that Huber's algorithm yields the best estimate for every case. The regular LS is able to somewhat reduce the attitude error in the presence of outliers, but diverges in the case of translation. Interestingly, all the IRLS methods that estimate the scale perform worse than the case where the scale is ignored. These results show the impact on the solution of proper scale estimation and the preference of Huber's algorithm over others. This suggests that robust estimation should be initiated with Huber's algorithm until convergence; to ensure that the rejection of outliers is maximized, some additional iterations can be performed with IRLS and a hard redescender, such as Tukey's function, using the (fixed) previously obtained estimate of σ, as suggested in Ref. [80].

VII. Filtering

A. Rigid Body Kinematics

The kinematics equation for $SE(3)$ in matrix form is [57]:

$$\dot{\boldsymbol{T}}_{B/T} = \boldsymbol{\varpi}^{B\wedge}_{B/T}\boldsymbol{T}_{B/T}, \quad \text{with } \boldsymbol{\varpi}^{B}_{B/T} \coloneqq \begin{pmatrix} \boldsymbol{\nu}^{B}_{B/T} \\ \boldsymbol{\omega}^{B}_{B/T} \end{pmatrix}, \tag{35}$$

where $\boldsymbol{T}_{B/T}$ is the rigid body pose mapping $\mathcal{F}_{\mathcal{T}}$ to $\mathcal{F}_{\mathcal{B}}$, and $\boldsymbol{\varpi}^{B}_{B/T}$ is the rigid body velocity of $\mathcal{F}_{\mathcal{T}}$ with respect to $\mathcal{F}_{\mathcal{B}}$ expressed in $\mathcal{F}_{\mathcal{B}}$. It is the concatenation of two terms: $\boldsymbol{\omega}$, the instantaneous angular velocity of the target as seen from

the chaser; and $\boldsymbol{\nu}$, the velocity of the point in $\mathcal{F}_{\mathcal{T}}$ that corresponds instantaneously to the origin of $\mathcal{F}_{\mathcal{B}}$. Dropping the subscripts and superscripts for succinctness, Eq. (35) is a first-order ordinary differential equation, and hence admits a closed-form solution of the form:

$$\boldsymbol{T}(t) = \exp((t - t_0)\boldsymbol{\varpi}^{\wedge})\boldsymbol{T}(t_0). \tag{36}$$

Equation (36) has the same form as Eq. (9), implying that $\boldsymbol{\varpi}$ is an element of $\mathfrak{se}(3)$. In agreement with the previous sections, this fact suggests that uncertainty can be introduced in the pose kinematics by modeling it as a local distribution in $\mathfrak{se}(3)$. As such, it is of interest to develop perturbation equations in terms of the kinematics in $\mathfrak{se}(3)$ so that these can be included as additive noise in a filtering scheme.

Following the approach of Ref. [86], the first two terms of Eq. (3) are used to linearize Eq. (9) as $\boldsymbol{T'} \approx (\boldsymbol{I} + \delta\boldsymbol{\xi}^{\wedge})\boldsymbol{T}$, where $\boldsymbol{T}$ is the nominal pose, $\delta\boldsymbol{\xi}$ is a small perturbation in $\mathfrak{se}(3)$, and hence $\boldsymbol{T'}$ is the resulting perturbed pose. Since $\boldsymbol{\varpi} \in \mathfrak{se}(3)$, this generalized velocity can be written directly as the sum of a nominal term with a small perturbation $\boldsymbol{\varpi'} = \boldsymbol{\varpi} + \delta\boldsymbol{\varpi}$. Substituting in Eq. (35), one has:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\left(\boldsymbol{I} + \delta\boldsymbol{\xi}^{\wedge}\right)\boldsymbol{T}\right) \approx (\boldsymbol{\varpi} + \delta\boldsymbol{\varpi})^{\wedge}\left(\boldsymbol{I} + \delta\boldsymbol{\xi}^{\wedge}\right)\boldsymbol{T}. \tag{37}$$

Expanding, ignoring the product of small terms and applying the Lie bracket of $\mathfrak{se}(3)$ yields the perturbation kinematics equation for $SE(3)$:

$$\delta\dot{\boldsymbol{\xi}} = \mathrm{ad}(\boldsymbol{\varpi})\delta\boldsymbol{\xi} + \delta\boldsymbol{\varpi}, \tag{38}$$

which is linear in both $\delta\boldsymbol{\xi}$ and $\delta\boldsymbol{\varpi}$.

B. Extended Kalman Filter Formulation

1. Motion Model

Equation (38) describes effectively the linearization of the rigid body kinematics around a nominal pose. Since it is defined with respect to elements of $\mathfrak{se}(3)$, perturbations in the motion can be modelled stochastically in terms of a local distribution (Section VI). The mean of this distribution may be injected into the nominal values via the exponential map. Under the assumption of Gaussian noise, this equation can therefore be regarded as the first step in defining an error-state to model how the motion evolves in time in the framework of an extended Kalman filter.

The kinematics of the target's motion with respect to the chaser spacecraft are correctly modelled by Eqs. (35) and (38). Modelling the relative dynamics, however, is not a clear-cut task. In the case of an asteroid mission, for example, the chaser could be considered to be inside the sphere of influence of the target and then Newton's second law of motion and Euler's rotation equation could be applied. However, in the case where both chaser and target are under

the influence of the same primary, the relative dynamics cannot be shaped as such.

In order to design a filter exclusively with relative states, and inspired by the method of Ref. [87], a broader constant generalized velocity motion model is adopted:

$$\dot{\boldsymbol{\varpi}}(t) = \boldsymbol{\eta}_{\boldsymbol{\varpi}}(t), \quad \boldsymbol{\eta}_{\boldsymbol{\varpi}}(t) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Q}(t)\delta(t-\tau)), \tag{39}$$

where it is assumed that the spectral density matrix $\boldsymbol{Q}(t)$ is diagonal. Note that, as stated in Ref. [87], this model does not assume that the chaser moves at a constant velocity over the entire sequence, but instead that undetermined accelerations with a Gaussian profile are expected to occur on average. In other words, one assumes that sizeable (relative) accelerations are unlikely to be experienced, which is a valid expectation for a space rendezvous.

Integrating Eq. (39) yields $\boldsymbol{\varpi}(t) = \boldsymbol{\varpi}(t_0) + \int_{t_0}^{t} \boldsymbol{\eta}_{\boldsymbol{\varpi}}(\tau)\,\mathrm{d}\tau$. The relation $\boldsymbol{\varpi}' = \boldsymbol{\varpi} + \delta\boldsymbol{\varpi}$ was assumed earlier, meaning that one can admit

$$\delta\boldsymbol{\varpi}(t) = \int_{t_0}^{t} \boldsymbol{\eta}_{\boldsymbol{\varpi}}(\tau)\,\mathrm{d}\tau. \tag{40}$$

Defining the error state $\delta\boldsymbol{x} \coloneqq (\delta\boldsymbol{\xi}, \delta\boldsymbol{\varpi})^\top$, the continuous-time error kinematics are written directly:

$$\frac{\mathrm{d}}{\mathrm{d}t}\delta\boldsymbol{x}(t) = \boldsymbol{F}(t)\delta\boldsymbol{x}(t) + \boldsymbol{G}(t)\boldsymbol{w}(t) = \begin{bmatrix} \mathrm{ad}(\boldsymbol{\varpi}) & \boldsymbol{I}_6 \\ \mathbf{0}_{6\times6} & \mathbf{0}_{6\times6} \end{bmatrix} \begin{pmatrix} \delta\boldsymbol{\xi} \\ \delta\boldsymbol{\varpi} \end{pmatrix} + \begin{bmatrix} \mathbf{0}_{6\times6} \\ \boldsymbol{I}_6 \end{bmatrix} \begin{pmatrix} \mathbf{0}_{3\times1} \\ \boldsymbol{\eta}_{\boldsymbol{\varpi}} \end{pmatrix}, \tag{41}$$

which shows that process noise is introduced in the system through the error generalized velocity vector. Equation (41) has the familiar solution [88]:

$$\delta\boldsymbol{x}(t) = \exp\left(\int_{t_0}^{t} \boldsymbol{F}(\tau)\mathrm{d}\tau\right)\delta\boldsymbol{x}(t_0) + \int_{t_0}^{t} \exp\left(\int_{s}^{t} \boldsymbol{F}(\tau)\mathrm{d}\tau\right)\boldsymbol{G}(s)\boldsymbol{w}(s)\mathrm{d}s \tag{42}$$

The error-state transition matrix has a known closed form [86]:

$$\boldsymbol{\Phi}(t,s) \coloneqq \exp\left(\int_{s}^{t} \boldsymbol{F}(\tau)\mathrm{d}\tau\right) = \begin{bmatrix} \mathrm{Ad}(\exp((t-s)\boldsymbol{\varpi}^\wedge)) & (t-s)\boldsymbol{B}((t-s)\boldsymbol{\varpi})) \\ \mathbf{0}_{3\times3} & \boldsymbol{I}_3 \end{bmatrix}, \quad \text{with } \boldsymbol{B}(\boldsymbol{\xi}) \coloneqq \begin{bmatrix} \boldsymbol{M}(\boldsymbol{\phi}) & \boldsymbol{N}(\boldsymbol{\xi}) \\ \mathbf{0}_{3\times3} & \boldsymbol{M}(\boldsymbol{\phi}) \end{bmatrix}, \tag{43}$$

where, defining $\phi \coloneqq \|\boldsymbol{\phi}\|$,

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{\xi}) \coloneqq{}& \frac{1}{2}\boldsymbol{\rho}^\wedge + \left(\frac{\phi - \sin\phi}{\phi^3}\right)\left(\boldsymbol{\phi}^\wedge\boldsymbol{\rho}^\wedge + \boldsymbol{\rho}^\wedge\boldsymbol{\phi}^\wedge + \boldsymbol{\phi}^\wedge\boldsymbol{\rho}^\wedge\boldsymbol{\phi}^\wedge\right) + \left(\frac{\phi^2 + 2\cos\phi - 2}{2\phi^4}\right)\left(\boldsymbol{\phi}^\wedge\boldsymbol{\phi}^\wedge\boldsymbol{\rho}^\wedge + \boldsymbol{\rho}^\wedge\boldsymbol{\phi}^\wedge\boldsymbol{\phi}^\wedge - 3\boldsymbol{\phi}^\wedge\boldsymbol{\rho}^\wedge\boldsymbol{\phi}^\wedge\right) \\ &+ \left(\frac{2\phi - 3\sin\phi + \phi\cos\phi}{2\phi^5}\right)\left(\boldsymbol{\phi}^\wedge\boldsymbol{\rho}^\wedge\boldsymbol{\phi}^\wedge\boldsymbol{\phi}^\wedge + \boldsymbol{\phi}^\wedge\boldsymbol{\phi}^\wedge\boldsymbol{\rho}^\wedge\boldsymbol{\phi}^\wedge\right). \end{aligned} \tag{44}$$

A closed-form of the discrete-time error process noise covariance matrix is found by directly solving the integral:

$$\boldsymbol{\Gamma}(t,s) \coloneqq \int_{t_0}^{t} \boldsymbol{\Phi}(t,s)\boldsymbol{G}(s)\boldsymbol{Q}(s)\boldsymbol{G}^{\top}(s)\boldsymbol{\Phi}^{\top}(t,s)\,\mathrm{d}s, \tag{45}$$

The derivation is monotonous but a matter of integrating each matrix element. To simplify it, the small angle approximation is applied and terms of $\mathcal{O}((t-s)^4)$ are discarded; the final result is not presented here.

2. Measurement Model

The correction stage of the EKF admits pseudo-measurements of the relative pose $y_i \in \mathcal{Y} \cong SE(3)$ as obtained through the refinement scheme of visual features correspondence from Section VI. These pseudo-measurements are acquired at each sampling time and modelled as being corrupted by a zero-mean white Gaussian noise term. One can thus write directly in discrete-time and matrix form:

$$\boldsymbol{Y} = \exp(\boldsymbol{\eta}_y^{\wedge})\boldsymbol{T}, \quad \boldsymbol{\eta}_y \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}), \tag{46}$$

where $\boldsymbol{Y} \in SE(3)$ is the matrix form of y. To linearize Eq. (46), similarly to the motion model, the elements of $SE(3)$ are rewritten as a small perturbation around a nominal term, i.e. $\boldsymbol{Y'} = \exp(\delta\boldsymbol{y}^{\wedge})\boldsymbol{Y}$, $\boldsymbol{T'} = \exp(\delta\boldsymbol{\xi}^{\wedge})\boldsymbol{T}$, and the exponential map is approximated by its first-order expression. Replacing in Eq. (46), expanding and neglecting the product of small terms, the following linearized relationship is obtained:

$$\boldsymbol{Y'} = \boldsymbol{T'}, \quad \delta\boldsymbol{y} = \delta\boldsymbol{\xi} + \boldsymbol{\eta}_y. \tag{47}$$

The full linearized measurement model is therefore:

$$\delta\boldsymbol{y}_i = \boldsymbol{H}\delta\boldsymbol{x} + \boldsymbol{\eta}_{y_i} = \begin{bmatrix}\boldsymbol{I}_6 & \boldsymbol{0}_{6\times 6}\end{bmatrix}\begin{pmatrix}\delta\boldsymbol{\xi} \\ \delta\boldsymbol{\varpi}\end{pmatrix} + \boldsymbol{\eta}_{y_i}, \quad \boldsymbol{\eta}_{y_i} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_i) \tag{48}$$

The covariance matrices of each pseudo-measurement are obtained as a product of the minimization scheme. In the case of the structural model constraints, the Jacobians $\boldsymbol{J}_s$ are of rank 6, so the covariance of the solution is given by backpropagation of the visual feature correspondences' own covariance [71]:

$$\boldsymbol{R}_s = \boldsymbol{\Sigma}_s = \left(\boldsymbol{J}_s^{\top}\boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{s}}\boldsymbol{J}_s\right)^{-1}, \tag{49a}$$

with $\boldsymbol{\Sigma}_{\boldsymbol{z},\boldsymbol{s}} = \sigma_{\boldsymbol{z},\boldsymbol{s}}^2\boldsymbol{I}$ and $\sigma_{\boldsymbol{z},\boldsymbol{s}}$ is obtained via M-estimation (cf. Subsection VI.D). The EKF innovation term at $t = k$ is:

$$\boldsymbol{\upsilon}_{s,k} = y_{s,k} \ominus \hat{u}_k^{-} \in \mathfrak{se}(3), \tag{49b}$$

where $\hat{u}_k^-$ is the predicted pose at $t = k$.

3. Measurement Gating

An additional step is employed prior to the correction to ensure the accurate functioning of the filter. This involves subjecting the incoming measurements to a validation gate, thus discarding potential spurious data. The validation gate is a threshold on the root mean square error (RMSE) of the residuals $\boldsymbol{r}$ as obtained by the M-estimation:

$$\mathrm{RMSE}(\boldsymbol{r}) \coloneqq \sqrt{\frac{\sum_{i=1}^{n} r_i^2}{n}}. \tag{50}$$

The RMSE provides an objective and clear interpretation of how close, in pixels, does the feature matching agree with the estimate of the pose.

C. Manifold State Prediction and Correction

The nominal state is construed as $\boldsymbol{x}^\top = (u^\top, \boldsymbol{\varpi}^\top)$, with $u \in \mathcal{U} \cong SE(3)$ representing the relative pose mapping $\mathcal{F}_\mathcal{T} \to \mathcal{F}_\mathcal{B}$ and $\boldsymbol{\varpi} \in \mathbb{R}^6$ is the generalized velocity satisfying the kinematics equation for $SE(3)$, Eq. (35). The nominal state estimate is updated with a linearized error state estimate $\delta\boldsymbol{x}^\top = (\delta\boldsymbol{\xi}^\top, \delta\boldsymbol{\varpi}^\top) \in \mathfrak{se}(3) \times \mathbb{R}^6$ via pose composition (cf. Eq. (13)), ensuring that u remains an element of $\mathcal{U} \cong SE(3)$. The algorithm's equations are valid for any chosen representation of u provided the appropriate composition $\oplus$ is used. State prediction is performed as:

$$\hat{u}_k^- = \hat{u}_{k-1} \oplus \Delta t \hat{\boldsymbol{\varpi}}_{k-1}, \quad \hat{\boldsymbol{\varpi}}_k^- = \hat{\boldsymbol{\varpi}}_{k-1}. \tag{51}$$

The state correction is given by:

$$\hat{u}_k^+ = \hat{u}_k^- \oplus \delta\hat{\boldsymbol{\xi}}_k^+, \quad \hat{\boldsymbol{\varpi}}_k^+ = \hat{\boldsymbol{\varpi}}_k^- + \delta\hat{\boldsymbol{\varpi}}_k^+, \tag{52}$$

where $\delta\boldsymbol{x}_k^{-\top} = (\hat{u}_k^{-\top}, \hat{\boldsymbol{\varpi}}_k^{-\top})$, $\delta\boldsymbol{x}_k^{+\top} = (\hat{u}_k^{+\top}, \hat{\boldsymbol{\varpi}}_k^{+\top})$ are the a priori and a posteriori error states, respectively. The covariance is calculated using the standard EKF equations.

In terms of the parameterization of u, a possible choice would be taking $u \to \boldsymbol{T}$ directly, in which case the composition operation is given by Eq. (9). Alternatively, the unit quaternion is a popular choice for attitude parameterization due to its compact and singularity-free representation, particularly in aerospace applications. The exponential map $\mathfrak{so}(3) \to SU(2)$ has a simple closed form given by $\exp_q = (\boldsymbol{\phi}^\top \sin(\phi/2)/\phi, \cos\phi/2)^\top$, with $\phi = \|\boldsymbol{\phi}\|$. [89]. In this case, the state vector becomes, with some abuse of notation, $\boldsymbol{x}^\top = (u^\top, \boldsymbol{\varpi}^\top) = (\boldsymbol{t}^\top, \boldsymbol{q}^\top, \boldsymbol{\varpi}^\top)$, which has dimension 13×1.

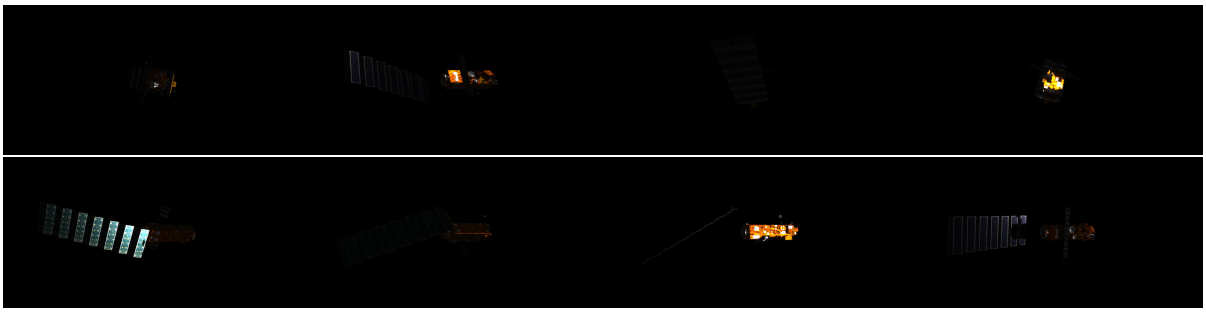


Fig. 8 Randomly sampled images from the BLENDER dataset.

VIII. Results

This section presents the the experimental results that validate the framework proposed herein. The considered target spacecraft is Envisat, a complex debris object, formed by several modules, namely a solar panel array, a synthetic aperture radar (SAR), and several antennae, among others, connected to a main body unit which is covered by multi-layer insulation (MLI). The experiments are structured into two main parts: i) evaluation of the coarse pose classification module, ii) evaluation of the fine pose estimation module, as initialized by the coarse.

All processing and simulations are carried out on a setup using an Intel® Core™ i7-6700 @ 3.40 GHz × 8 core processor, 16 GB RAM system.

A. Coarse Pose Classification

1. Simulations on BLENDER Dataset

The performance of the coarse pose classification module is first evaluated independently. To this end, a synthetic dataset was generating using a CAD model of Envisat freely available from the space simulator Celestia[‡]. The CAD model was modified using the open-source 3D computer graphics software Blender[§] with additional materials and textures in order to yield a realistic aspect in face of the expected low Earth orbit conditions. This includes a complete remodeling of the MLI to recreate diffuse light reflection, and the addition of reflective materials to the solar panel. Both of these aspects represent challenging imaging conditions in the visible wavelength for relative pose estimation. The dataset is accordingly designated as the BLENDER dataset.

BLENDER (Fig. 8) is generated by uniformly sampling the viewsphere (Section V, Fig. 3) with a mesh step Δ_{mesh} equal to 1 deg, yielding over 64 400 images in total and approximately 100 per class[¶]. For each rendered image, two illumination sources are added: a constant, uniform, low-intensity lighting emulating the Earth's albedo; and a high-intensity lighting with a randomly varying direction to emulate different Sun angles. All images feature a black, deep-space background, where the target is the only object present in the field-of-view (FOV). This is to allow for a

[‡] http://celestia.space/.

[§] http://www.blender.org/

[¶] On the poles (0 deg and 180 deg elevation) a change in azimuth only produces an in-plane rotation and not a change in viewpoint, leading to fewer renderings in these cases.

Table 1 Settings used for k-folds validation of the coarse pose classification module.

Parameter	Units	Value
Viewsphere azimuth mesh step, $\Delta^{\text{az}}_{\text{mesh}}$	[deg]	1
Viewsphere elevation mesh step, $\Delta^{\text{el}}_{\text{mesh}}$	[deg]	1
Viewsphere azimuth class step, $\Delta^{\text{az}}_{\text{class}}$	[deg]	10
Viewsphere elevation class step, $\Delta^{\text{el}}_{\text{class}}$	[deg]	10
Total classes	[-]	648
Folds, k	[-]	5

straightforward binary segmentation of the shape prior to the computation of the ZMs.

For the current analysis, images from the BLENDER dataset are grouped into bins of 10 deg in azimuth and elevation, thus defining the minimum achievable accuracy for the coarse pose classifier. This reduces the classification problem to 648 possible classes. To expand the size of the training population, image augmentation is performed. This consists in post-processing the renders by randomly applying some transformations; variations are introduced in terms of scale, in-plane rotation, perspective transforms, and binary segmentation threshold, yielding 500 images per class. To test the performance of the algorithm, a stratified k-fold cross-validation is then performed on a 80%–20% train-test split. The module was built in Matlab and the Bayesian classifier was implemented using the GMMBayes toolbox[‖]. Table 1 summarizes the used parameters for the cross-validation test.

In order to have a comparative insight of how the algorithm performs, two competitor methods are additionally benchmarked.The first competitor method is similar in the sense that it also relies on the ZM-based description of the target's shape and Bayesian classification, except that the likelihoods $p\left(\boldsymbol{y}|\mathcal{C}_m\right)$ are instead modeled with a single Gaussian (i.e. $n = 1$ mixture components in Eq. (18)). The second competitor method uses local features rather than a global ZM descriptor and is based on the bags-of-keypoints, or bags-of-visual-words (BoVW), method [90]. BoVW is an image classification and scene recognition algorithm inspired on the bags-of-words model from the field of natural language processing. First, feature extraction is performed on the images from each class (in this case, ORB+FREAK features). The full set of keypoints is then iteratively clustered into a pre-defined number of compact, non-overlapping groups, forming the visual vocabulary (the bag-of-keypoints) of the full domain. Each cluster center is therefore a word of the vocabulary. The features from each image are binned according to the cluster centers of the vocabulary, yielding one fixed-size histogram of visual word occurrences per image. The histograms belonging to each class are then used to train a classifier. Since binary keypoint descriptors are considered in this analysis, the clustering is performed using k-medoids rather than the typical k-means. Due to the large number of classes considered, rather than clustering the full set of features directly, 10 clusters are extracted for each class and then concatenated to form the global vocabulary as in [91]. For similar reasons, a Bayesian classifier is used rather than an SVM. Both competitor methods are also implemented in Matlab and trained with the same number of images.

[‖] http://www.it.lut.fi/project/gmmbayes/.

Table 2 Average expected attitude error e_{att} for the coarse pose classification on the BLENDER dataset.

Metric	Units	BoVW	Gaussian	GMM
Mean e_{att}	[deg]	76.86	14.21	9.35
Median e_{att}	[deg]	70.62	0.00	0.00

The results are illustrated in Fig. 9 for azimuth and elevation classification performance in terms of the probability mass function (PMF) for each class in the form of a histogram. The horizontal axis represents the error in terms of bin distance, where a value of "0" represents a correct classification. Note that the azimuth error is wrapped to a maximum equivalent error of 180 deg. An alternative representation of the errors is shown is Fig. 10 in terms of the (discrete) cumulative distribution function (CDF) for a better comparison of the three methods. The average expected combined attitude error is also showcased (Table 2); this is calculated by converting the azimuth and elevation from the center of each class into the equivalent predicted relative attitude quaternion $\hat{\boldsymbol{q}}$ (the roll angle is assumed the same for each class). The error quaternion $\delta\boldsymbol{q} = \hat{\boldsymbol{q}}^{-1} \otimes \boldsymbol{q}$ is then computed from the ground truth. Lastly, the rotation angle is obtained from δq, the scalar component of $\delta\boldsymbol{q}$, through the well-known [64] relation:

$$e_{\text{att}} = 2\arccos(\delta q_{C/T}). \tag{53}$$

The overall performance of the BoVW classifier is poor and vastly outperformed by the two shape-based methods. The correct classification rate for the azimuth is approximately 15%. A cumulative score of 50% is only achieved for a distance of 5 classes, i.e. a maximum average expected error of 45 deg. The results for the elevation classification are improved, which is expected, as it involves fewer classes (19 as opposed to 35 for the azimuth); this is the case for all benchmarked methods. However, the performance in this case remains far below the rest. The mean average expected attitude error is over 75 deg, making it unfit for coarse pose classification, and demonstrating that local features are not distinctive enough for the viewpoint classification of the spacecraft.

The performance of both shape-based classifiers is comparable. However, the details on Fig. 9 expose the presence of multiple modes in the case of the single Gaussian modeling. This phenomenon is more prominent for the azimuth error (also present on the BoVW) and stems from the ambiguous projected shape of the target when imaged from opposing viewpoints. There is an additional error peak around the 90 deg difference in azimuth; overall this profile is consistent with a target of cuboid shape, such as the main bus of Envisat. The effect is also observable, to an extent, for the elevation error. On the other hand, these peaks have practically been mitigated in the case of the GMM. The correct classification rate is 71.35% for azimuth and 75.86% for elevation. These represent gains of approximately 10% and 5%, respectively, comparatively to the single Gaussian modelling. In terms of average expected attitude error, this translates into an mean improvement of 5 deg. Overall, the GMM leads to 90.41% of the data being classified with a bin

Table 3 Mean computational execution times per image for the coarse pose classification on the BLENDER dataset.

Operation	Units	BoVW	Gaussian	GMM
Feature detection	[ms]	81.99	62.78	62.78
Inference	[ms]	87.75	545.58	138.93
Total	[ms]	169.73	608.36	201.71

distance less than or equal to 1, i.e. with a maximum expected error of 20 deg, and equivalently 92% for the elevation.

Table 3 displays the computational times of the three benchmarked methods. The feature detection cost is comparable for all; note that the feature detection is the same for both the single Gaussian and GMM classifiers. For these two, the inference cost is superior, particularly in the case of the former. However, this could be attributed to the fact that a different Matlab toolbox was used for the classification process of each. The total computational cost of the proposed GMM-based classifier is comparable to that of the BoVW, averaging 200 ms. This does not represent a significant bottleneck to the fine pose estimation module as it is only ran once, and the runtime is expected to decrease even more when implemented on a lower level programming language.

2. Simulations on SPEED Dataset

To offer some degree of comparison with the current state-of-the-art, the proposed coarse pose classification module is also tested on the publicly available Spacecraft PosE Estimation Dataset (SPEED) dataset, which was used to benchmark the entries of the ESA SPEC [54]. Overall, SPEED is a challenging dataset based on the Tango spacecraft (Fig. 11). This target is also characterized by a cuboid shape, albeit much more compact in comparison to Envisat, leading to more ambiguous viewpoints, which could prove to be limiting for shape-based classifiers. SPEED is composed of both synthetic and laboratory-acquired data divided into train (SPEED/TRAIN, SPEED/REAL) and test (SPEED/TEST, SPEED/REAL-TEST) sets, numbering 12 000, 5, 2998, and 300 images, respectively. Contrary to the train sets, the ground truth pose for the test ones is not made publicly available, and the only way to acquire a performance metric is by submitting the results on the SPEC website[**]. At the time of publication, the website featured a post-mortem version allowing the submission of results despite the ending of the challenge.

Since the absence of a ground truth would not allow for an in-depth analysis of the performance for the coarse pose classification pipeline, the method is tested exclusively on SPEED/TRAIN. Furthermore, the dataset contains some aspects related to the extraction of the target's shape that had to be adapted in order for the current analysis to be performed. Firstly, synthetic SPEED images partly contains images where the Earth is present in the background (Fig. 11a). The proposed pipeline does not include target/background segmentation, and therefore these images have been removed prior to testing. Secondly, all images are contaminated by Gaussian noise, making image binarization

[**] https://kelvins.esa.int/satellite-pose-estimation-challenge.

Table 4 Average expected attitude error e_{att} for the coarse pose classification on the SPEED dataset.

Metric	Units	Value
Mean e_{att}	[deg]	34.81
Median e_{att}	[deg]	10.00

non-trivial, which leads to noisy extracted shapes and more than often to deficient segmentations, which will affect the quality of the classifier. As such, prior to binarization, the images have to be pre-processed in an impromptu way, under which some errors still remain (Fig 11c). This is not meant to be an optimal process, and future work will include the development of a target segmentation module to cope with these limitations.

Similarly to the analysis for BLENDER, a k-folds cross validation is performed on SPEED/TRAIN using same settings from Table 1. The results for the proposed method in terms of the classification PMF are presented in Fig. 12. The average expected attitude error is shown in Table 4. As expected, the benchmarked performance is inferior to that attained for the BLENDER dataset. Notably, there is a clear presence of a second azimuth mode at 180 deg which is not fully tackled by the GMM. Additionally, it can be seen that the elevation PMF tail also flattens out at a higher value. The correct classification rates for azimuth and elevation are approximately 50% and 60%, respectively. However, 75% of the data azimuth accuracy is concentrated at a bin distance less than or equal to 1, equivalently 85% for the elevation. The overall mean attitude error is situated at 34.81 deg. This represents a slightly lower performance than the current state-of-the-art deep learning methods (see Section II), but despite not explicitly tackling the target segmentation problem, the proposed classifier still obtains good enough results to be used as an initialisation method, as it is meant, to the fine estimation module, at a fraction of the required training and inference computational times.

B. Fine Pose Estimation

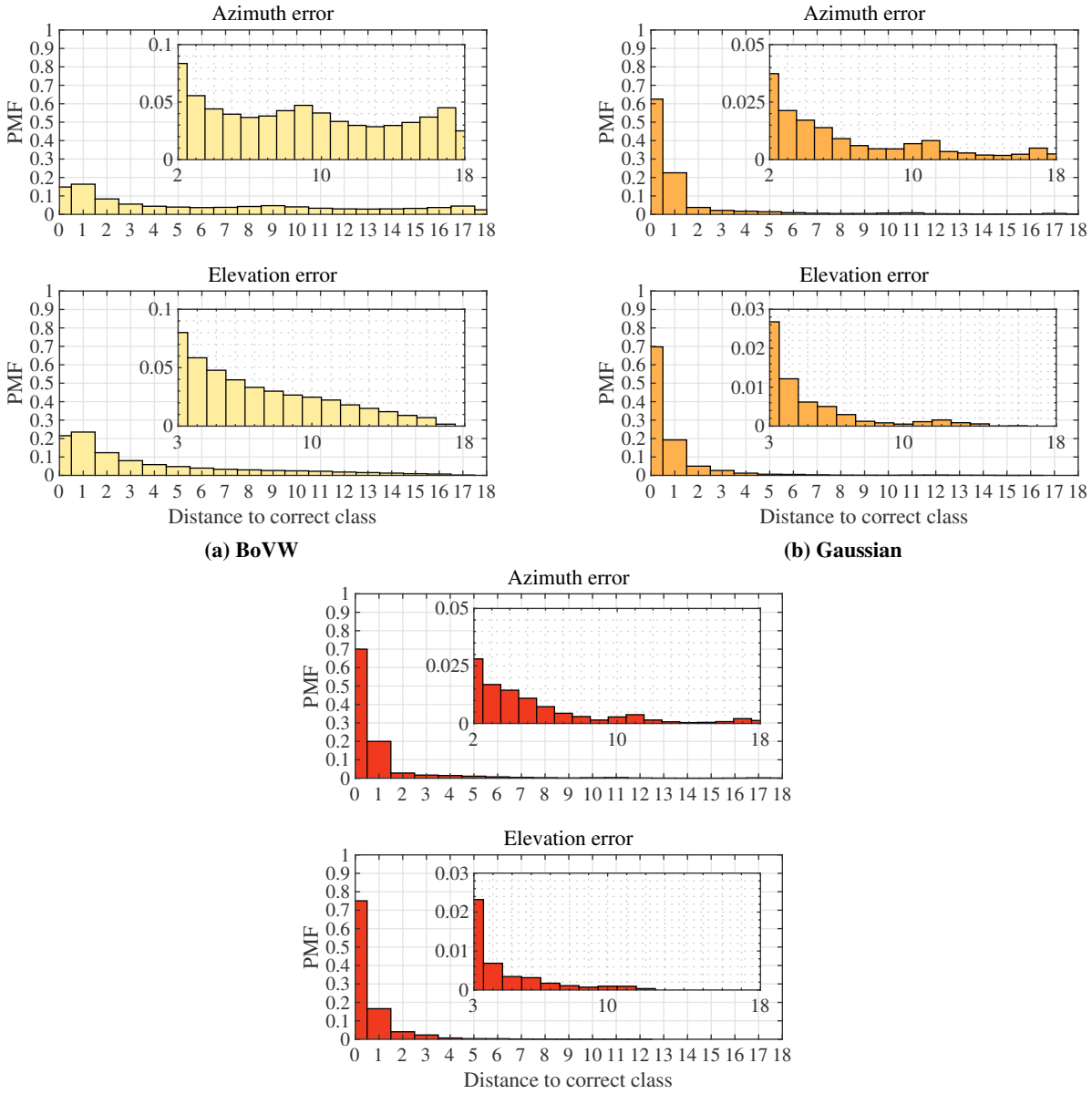
1. Simulations on ASTOS Dataset

In this section, the performance of the full spacecraft relative pose estimation pipeline is assessed. This includes the initialization procedure with the coarse pose classifier followed by the pose refinement using local features (cf. Section IV). The pipeline is first tested with synthetically generated rendezvous trajectories. To this end, the Astos Camera Simulator[††] was used, which is capable of recreating the true relative states of the target and chaser spacecraft, the Sun, and Earth, emulating the realistic, challenging lighting conditions that are to be expected in-orbit. The orbit of Envisat was simulated using the two-line element set corresponding to 30 October 2017[‡‡]. This corresponds to the orbital parameters in Table 5.

In terms of rendezvous kinematics, two different trajectories are considered (Fig. 13): ASTOS/01 considers the

[††] http://www.astos.de/.

[‡‡] TLE data obtained from NORAD Two-Line Element Sets Current Data at http://www.celestrak.com/NORAD/elements/.



(a) BoVW **(b) Gaussian** **(c) GMM**

Fig. 9 Histogram of results of the k-folds validation for the coarse pose classification on the BLENDER dataset.

Table 5 Characterization of the chaser and target's orbital motion for rendezvous simulation on the ASTOS dataset.

Parameter	Units	Value
Target orbital parameters		
Semi-major axis	[km]	7.1427×10^3
Eccentricity	[-]	7.6112×10^{-4}
Inclination	[deg]	98.2164
Right ascention of the ascending node	[deg]	343.0760
Argument of perigee	[deg]	189.5264
True anomaly	[deg]	3.0109

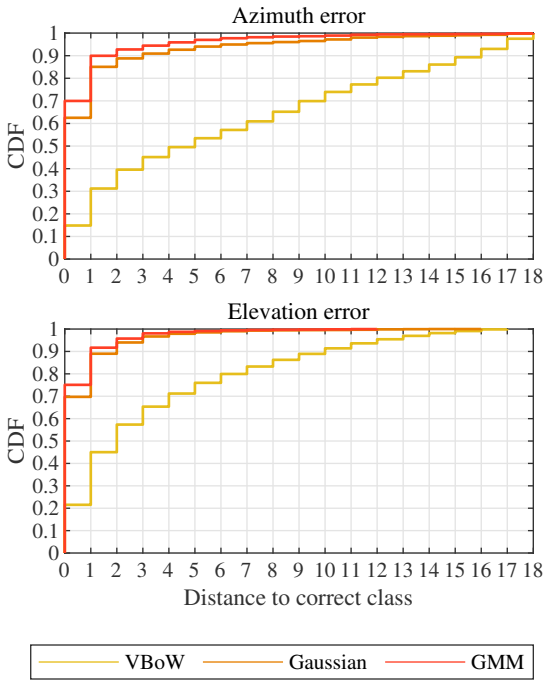
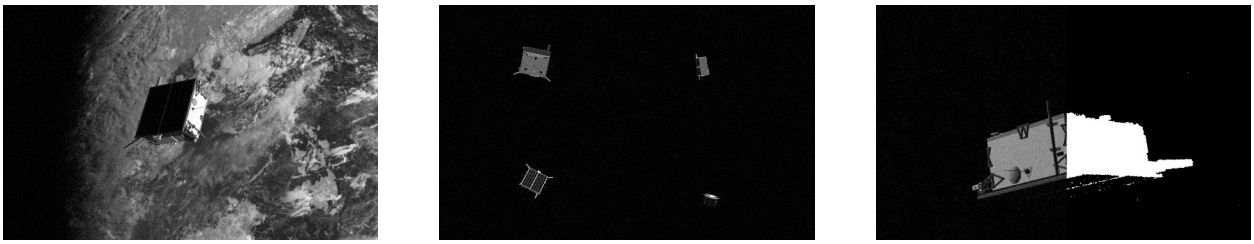


Fig. 10 Cumulative performance of the k-folds validation for the coarse pose classification on the BLENDER dataset.



(a) Earth in the background **(b) Ambiguous shape** **(c) Low lighting and noise**

Fig. 11 Characteristics of the SPEED dataset, sampled from SPEED/TRAIN.

chaser observing the target from a hold point on V-bar in the target's local-vertical local-horizontal (LVLH) frame at a distance of 50 m; ASTOS/02 emulates a forced translation along V-bar with a cross-track amplitude of 1 m, beginning at a distance of 100 m and ending at a distance of 50 m. The evolution of the rotational states are displayed in Fig. 14 for each trajectory, in the form of the three scaled components of the relative axis-angle vector $\boldsymbol{\phi}_{C/T}$. The tumbling mode of ASTOS/01 consists of the spin axis in the target frame aligned with the $+\boldsymbol{t}_2$ axis, the spin axis in the LVLH frame aligned with the +H-bar axis, and a tumbling rate of 3.5 deg/s. The tumbling mode of ASTOS/02 consists of the spin axis in the target frame along a direction contained in the $\boldsymbol{t}_2$-$\boldsymbol{t}_3$ plane at 45 deg, the spin axis in the LVLH frame aligned with the +H-bar axis, and a tumbling rate of 5 deg/s.

Table 6 shows the parameters employed in the test. The compact mvBlueFOX MLC202b camera is simulated, as it is similar to the one utilized in the experimental setup (cf. Section VIII.B.2). A step of 9 deg both in azimuth and

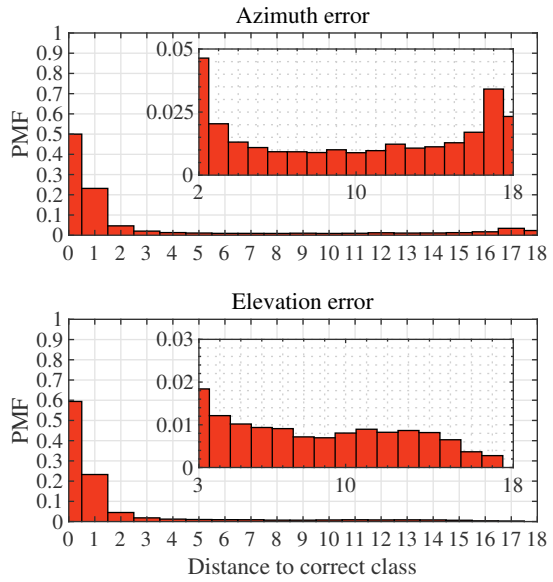
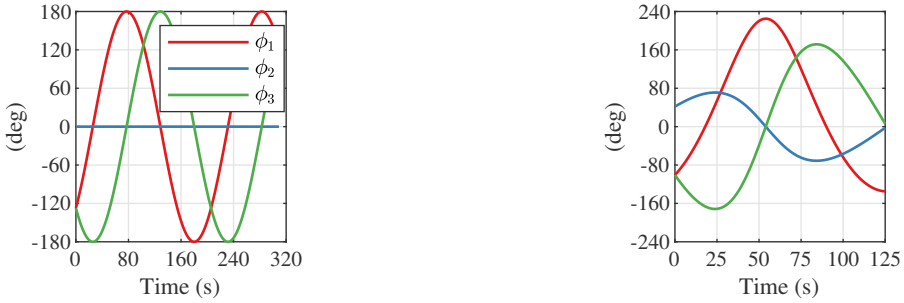


Fig. 12 Histogram of results of the k-folds validation for the coarse pose classification on the SPEED dataset.

(a) ASTOS/01 **(b) ASTOS/02**

Fig. 13 Simulated relative trajectories for the ASTOS dataset on the target LVLH frame.



(a) ASTOS/01 **(b) ASTOS/02**

Fig. 14 Simulated relative attitude states for the ASTOS dataset of the target frame with respect to the chaser frame.

Table 6 Pose estimation pipeline configuration and numerical settings.

Parameter	Units	Value (ASTOS)	Value (UASL)
Camera			
Focal length	[mm]	5	3
Resolution	[px]	640×480	752×480
FOV	[deg]	53.06×36.82	79.52×58.03
Framerate	[Hz]	10	10
Viewsphere			
Azimuth step	[deg]	9	9
Elevation step	[deg]	9	9
Total keyframes	[-]	800	800

elevation was chosen to build the offline database, resulting in 800 keyframes. Note that, for this particular simulated trajectory, only 40 keyframes would be required, since the azimuth is fixed; however, to stress the algorithm, the full set of possible keyframes to choose from is kept. The EKF is run with a timestep of 0.1 s (the sampling rate of the camera). The initial filter pose state $\hat{\boldsymbol{t}}_0, \hat{\boldsymbol{q}}_0$ is initialized with the result of the coarse pose estimation, while the velocity state is pessimistically assumed to be equal to zero. The initial covariance $\hat{\boldsymbol{P}}_0$ and the process noise covariance $\sigma_v^2, \sigma_\omega^2$ are tuned empirically, whereas the measurement noise covariance is automatically determined via M-estimation.

Four different methods are compared: i) EPnP with feature point matches [70] and RANSAC for outlier rejection; ii) the method developed in the authors' previous paper [38], using M-estimation fusing point and edge features; iii) the model-free ORB-SLAM2 [18]; and iv) the framework proposed in this paper. In the case of the first two methods, the next keyframe is determined by the pose estimated in the previous time-step. As ORB-SLAM2 is a model-free method, the first built keyframe is arbitrarily oriented and scaled; as such, for the context of this analysis, it is scaled with the corresponding trajectory ground truth. The position error is calculated according to the L2 norm:

$$e_{\text{pos}} = \|\boldsymbol{t}_{C/T} - \hat{\boldsymbol{t}}_{C/T}\|_2. \tag{54}$$

Analogously, the velocity errors are computed the same way. The attitude error is computed according to Eq. (53).

The framework was coded in the C++ programming language, whereas the OpenCV library[§§] was used for IP-related functions.

The results of the relative pose estimation for ASTOS/01 are shown in Fig. 15. It can be seen that EPnP+RANSAC is not able to converge at all. The pure M-estimator yields a decent estimate for the first few frames, but the error quickly begins to grow until the algorithm diverges completely at $t = 5$ s. ORB-SLAM2 is only capable of providing a pose estimate for three segments of the trajectory, corresponding to the parts where the SAR-side is facing the camera and the number of keypoints is maximal and relatively stable. Nevertheless, the estimate quickly drifts in the case of the position,

[§§] https://opencv.org/.

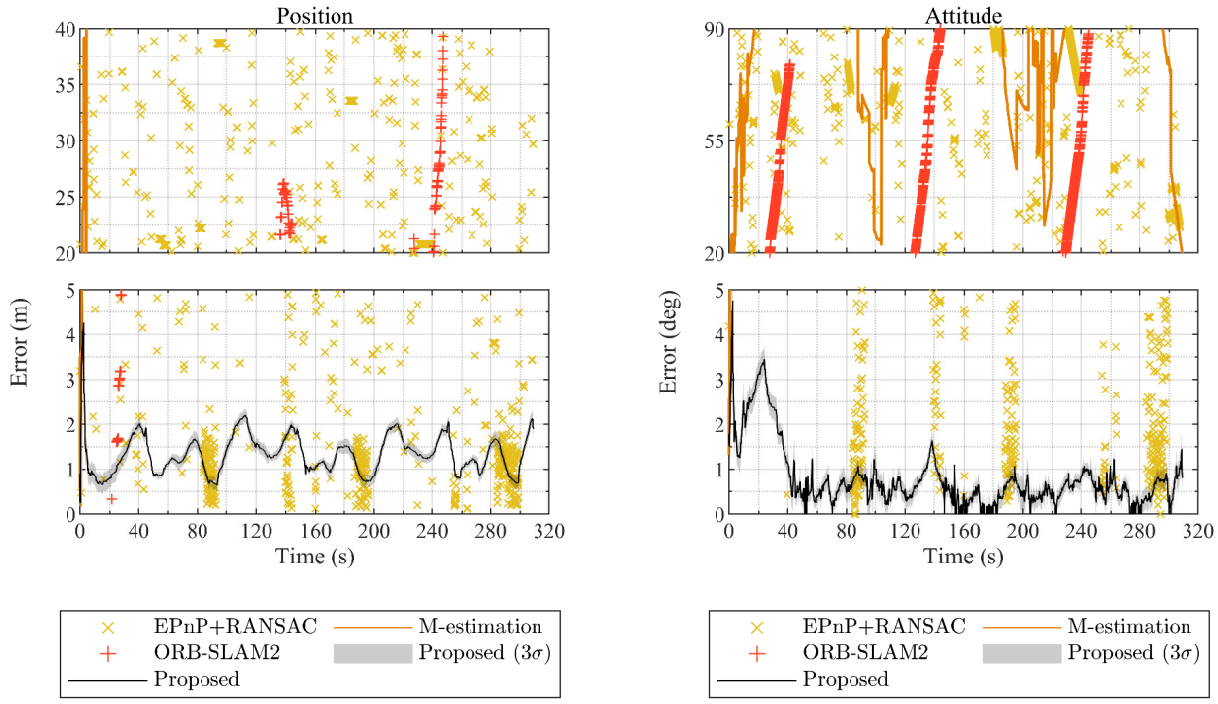


Fig. 15 Nominal pose estimation errors for the ASTOS/01 trajectory.

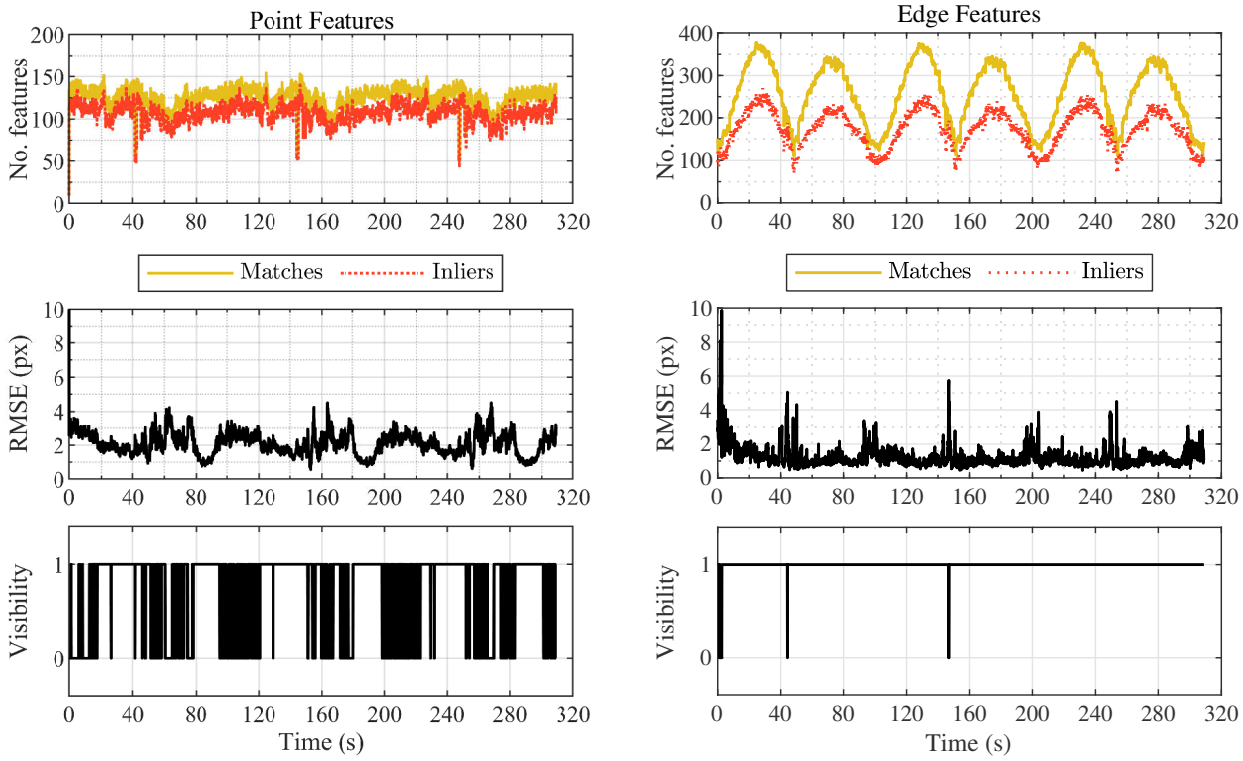


Fig. 16 Feature statistics for nominal pose estimation sequence of the ASTOS/01 trajectory.

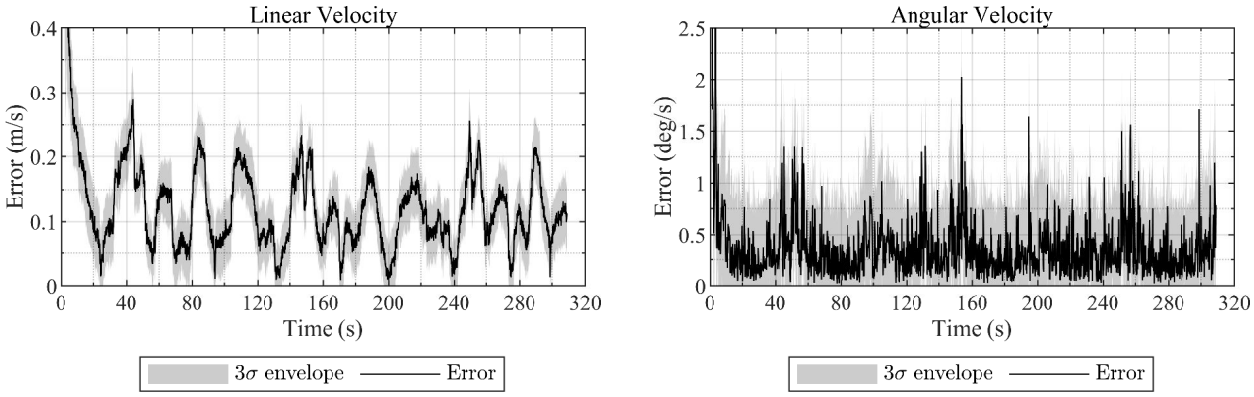


Fig. 17 Nominal velocity estimation errors for the ASTOS/01 trajectory for the proposed framework.

(a) Initial pose at $t = 0\,\mathrm{s}$ **(b)** $t = 50\,\mathrm{s}$ **(c)** $t = 115\,\mathrm{s}$ **(d)** $t = 285\,\mathrm{s}$

Fig. 18 Qualitative results of the relative pose estimation for the ASTOS/01 dataset. The edges of the SAR and solar panel are reprojected in green using the estimated pose.

and it is entirely wrong in the case of the attitude. On the other hand, the proposed framework converges at around $t = 10\,\mathrm{s}$. The steady state error is bounded at approximately 2 m for position, which corresponds to 4% of the range distance, whereas the attitude error is bounded at 1.5 deg. Figure 16 exhibits some figures of merit pertaining to the point and edge features in the simulation run, namely the number of matches and inliers, the RMSE of the M-estimation, and the feature visibility with respect to the validation gating applied prior to the filtering. A threshold of 2.5 px was applied for the points and 5 px for the edges. The number of matches fluctuates more in the case of edges; this is due to the relative circular trajectory in which the imaging area of the target changes. The peaks correspond to the sections where the $\boldsymbol{t}_1 - \boldsymbol{t}_2$ plane $\in \mathcal{F}_\mathcal{T}$ is imaged by the chaser, whereas the valleys correspond to an imaging of the $\boldsymbol{t}_2 - \boldsymbol{t}_3$ plane. However, the RMSE of the point features is on average greater than that of the edges, which results in fewer periods of visibility for the former. The periods of higher RMSE correspond to images of the $\boldsymbol{t}_1 - \boldsymbol{t}_2$ plane, where the image of the target is dominated by the MLI coverage and the solar panel. Despite this, the guided feature matching algorithm prevents the point features' visibility from being constantly null during these periods.

The relative velocity estimation errors as output by the filter are also shown (Fig. 17). The linear velocity steady-state error does not exceed 0.3 m/s, whereas the angular velocity is bounded at 2 deg/s. The latter quantity is much noisier than the former, since there are two out-of-plane dimensions, compared to one for the linear velocity, highlighting the challenge of depth estimation with a monocular setup. Lastly, Fig. 18 illustrates the some frames of the synthetic dataset with the estimated pose superimposed.

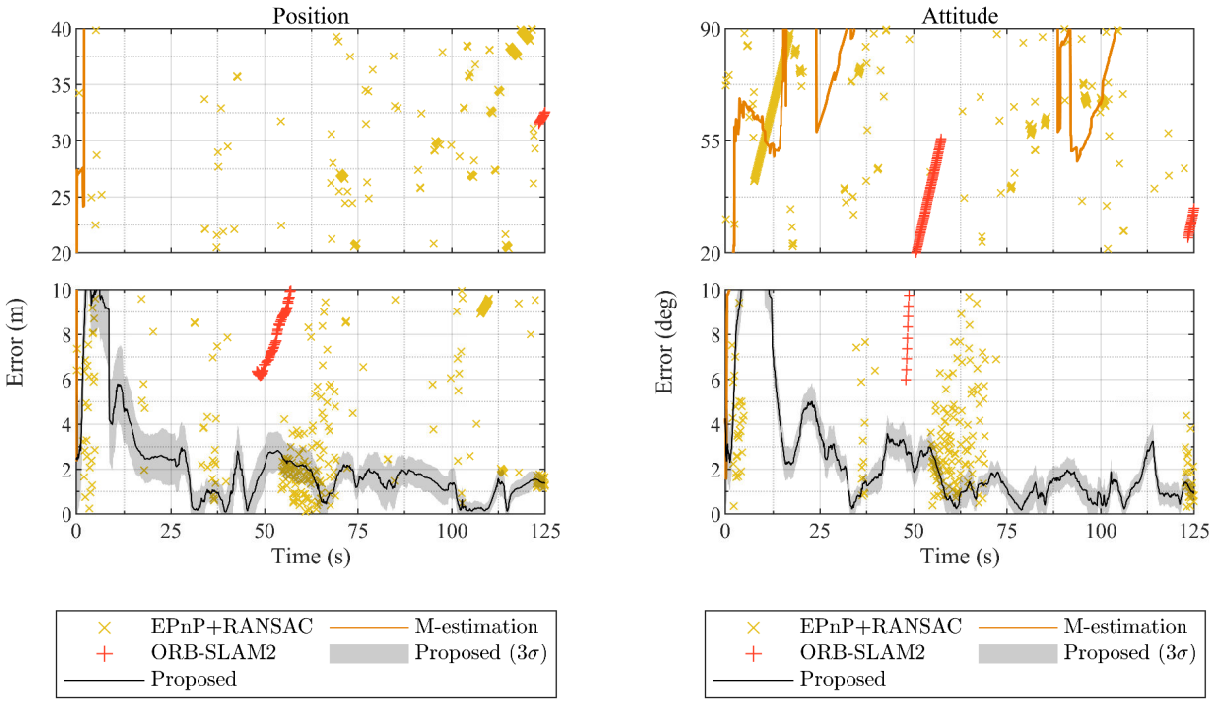


Fig. 19 Nominal pose estimation errors for the ASTOS/02 trajectory.

Table 7 Mean computational execution times per image for the fine pose estimation on the ASTOS dataset.

EPnP+RANSAC	M-estimation	ORB-SLAM2	Proposed	Units
63.32	72.81	20.00	127.72	[ms]

The pose estimation results for the ASTOS/02 trajectory are portrayed in Fig. 19. This trajectory is considered to be more challenging due to the change of position depth and more complex tumbling. EPnP+RANSAC and the pure M-estimator, again, quickly diverge. ORB-SLAM2 is now only capable of briefly providing a solution for two sections and with unsatisfactory quality. The proposed method takes slightly longer to converge (at around t =15 s). From the 3σ standard deviation envelope estimated by the filter, the uncertainty is noticeably higher comparatively to ASTOS/01. The steady state position error behaves similarly, not exceeding 4% of the range. The attitude error is slightly worse, reaching a peak of 5 deg right after the transient, but remaining bounded at 3.8 deg from thereon. The introduced variations in the relative motion drive the velocity errors considerably higher (Fig. 20): while the linear velocity error appears to decrease along with the range, the angular velocity error converges to a steady state value of 4 deg/s. In spite of this, the predictive matching module that relies on the EKF prediction remains robust enough to provide an accurate solution of the pose. Qualitative results are exhibited in Fig. 21.

The mean computational cost per image of the four methods is benchmarked in Table 7. ORB-SLAM2 is by far the fastest method. Note, however, that for most of the benchmarking, the algorithm was unable to initialize, and thus the

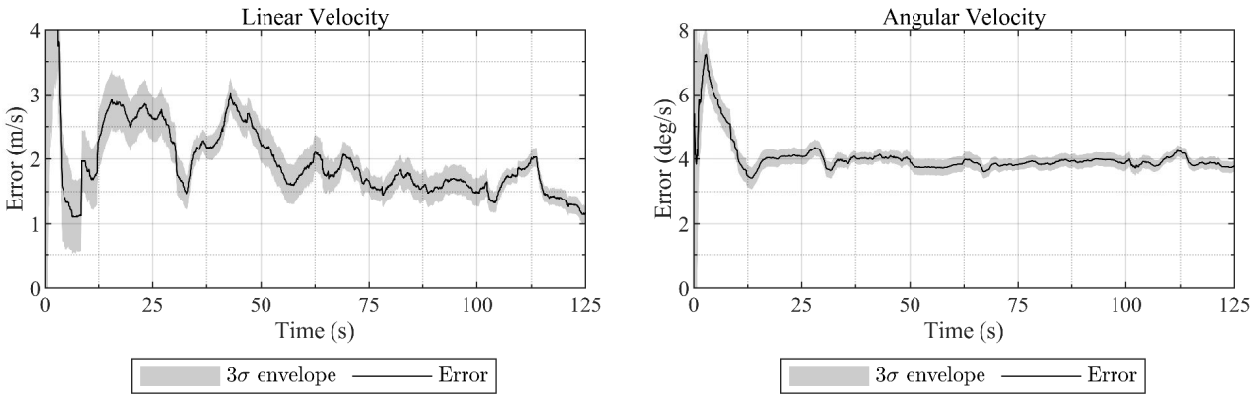
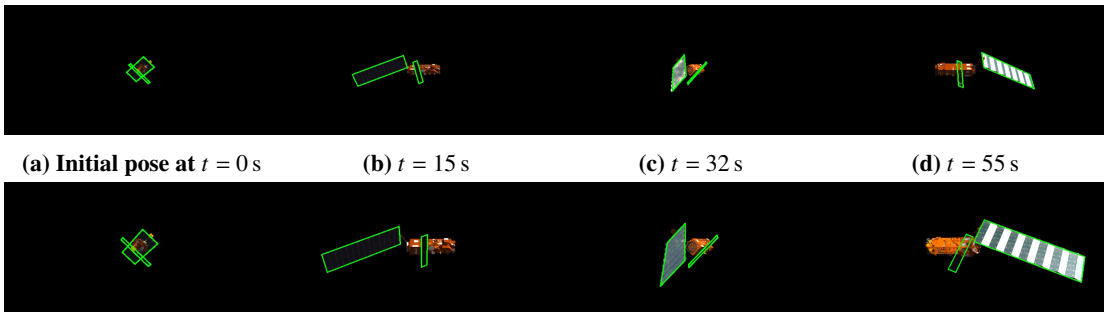


Fig. 20 Nominal velocity estimation errors for the ASTOS/02 trajectory for the proposed framework.



(a) Initial pose at $t = 0$ s **(b)** $t = 15$ s **(c)** $t = 32$ s **(d)** $t = 55$ s

(e) $t = 72$ s **(f)** $t = 90$ s **(g)** $t = 104$ s **(h)** $t = 120$ s

Fig. 21 Qualitative results of the relative pose estimation for the ASTOS/02 dataset. The edges of the SAR and solar panel are reprojected in green using the estimated pose.

majority of the heavy-lifting was avoided. Conversely, the proposed method (non-optimized code) exhibits the highest cost, which is broken down in Table 8. It is clear that the bottleneck resides in the feature extraction and M-estimation tasks. Notably, the computation of the FREAK descriptors takes approximately 40 ms, and the combined M-estimation routines take 55 ms to run; both make up almost 75% of the total runtime. By limiting the number of detected features, both of these figures of merit can be decreased on one go. Future work will include a trade-off analysis on the influence of limiting the number of features on the accuracy of the pipeline.

2. Validation on UASL Dataset

In this section, the framework is validated in laboratory. The experimental setup established in the Unmanned Autonomous Systems Laboratory (UASL) at Cranfield University is portrayed in Fig. 22. A scaled model of Envisat was built with 1-DOF in rotational motion (along the $\boldsymbol{t}_1$ axis). Due to the dimensions of the mock-up, a custom-built LED illumination panel running at 900 W is used to simulate direct sunlight. The mvBlueFOX MLC200wC camera with a 3 mm lens is used to acquire the dataset. The initial position and attitude are, respectively, $\boldsymbol{t}^{\mathcal{C}}_{\mathcal{C}/\mathcal{T},0} = (-0.1418, -0.0449, 1.9312)^\top$ m and $\boldsymbol{q}_{\mathcal{C}/\mathcal{T},0} = (-0.0356, -0.0336, -0.01751, 0.9986)^\top$. The motion is

Table 8 Breakdown of mean computational execution times per image for the proposed fine pose estimation method on the ASTOS dataset.

Operation	Units	Value
Point detection	[ms]	7.84
Point description	[ms]	39.14
Image binarization	[ms]	0.15
Edge detection	[ms]	2.54
Point matching	[ms]	0.53
Edge matching	[ms]	2.32
Point M-estimation	[ms]	7.83
Edge M-estimation	[ms]	56.98
Keyframe selection	[ms]	10.24
EKF prediction	[ms]	0.06
EKF correction	[ms]	0.09
Total	[ms]	127.72

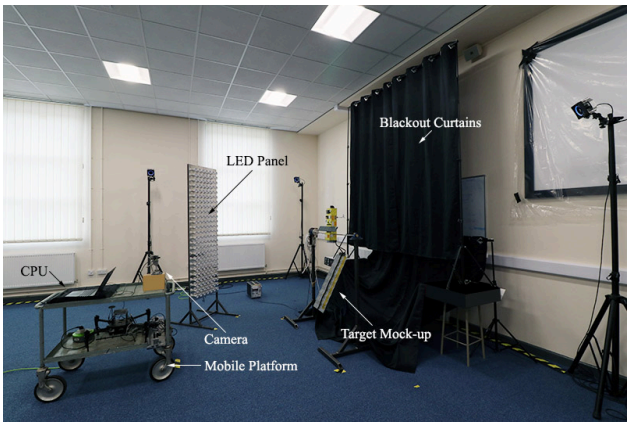


Fig. 22 Setup for laboratory validation.

analogous to the simulated one for ASTOS/01 (cf. Fig. 13) and the rotation rate is constant and equal to 5.73 deg/s. The Envisat mockup is modeled in Blender and textured with real images to generate the offline keyframe database. The ground truth is obtained by manually registering the first frame and by propagating the state using the rigid body's constant rotation rate from the static camera's viewpoint. It is assumed that the background can be subtracted correctly.

Figure 23 displays the pose estimation errors as achieved by the framework; Fig. 24 shows the velocity estimation errors; Fig. 25 depicts a set of frames from the lab sequence including initialization and reprojection of the pose. The magnitude of the attained errors is analogous to that obtained for the synthetic dataset: a maximum of 5% position estimation error with respect to the range, whereas the attitude error in steady-state does not exceed 2.5 deg (while remaining generally under 1.5 deg). As expected, the angular velocity error estimation is more noisy than the linear velocity one.

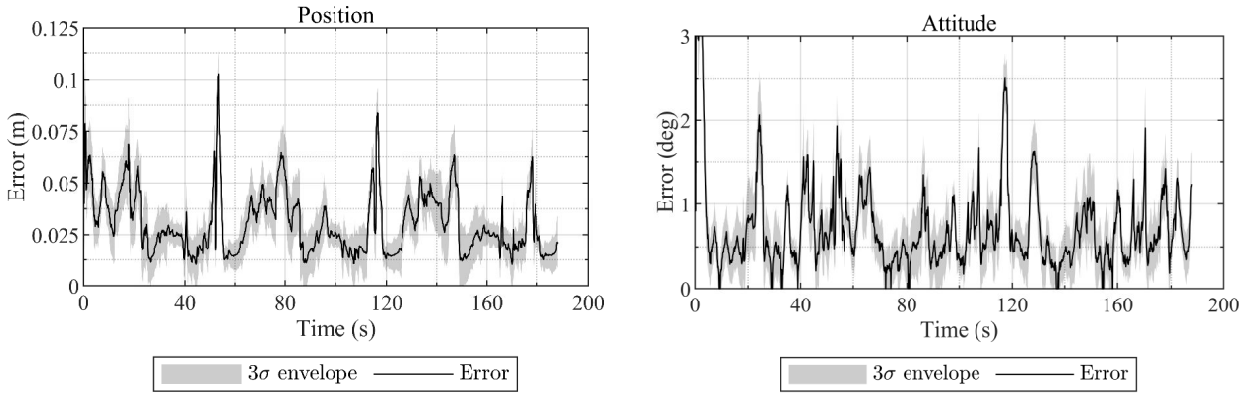


Fig. 23 Pose estimation errors for the UASL dataset.

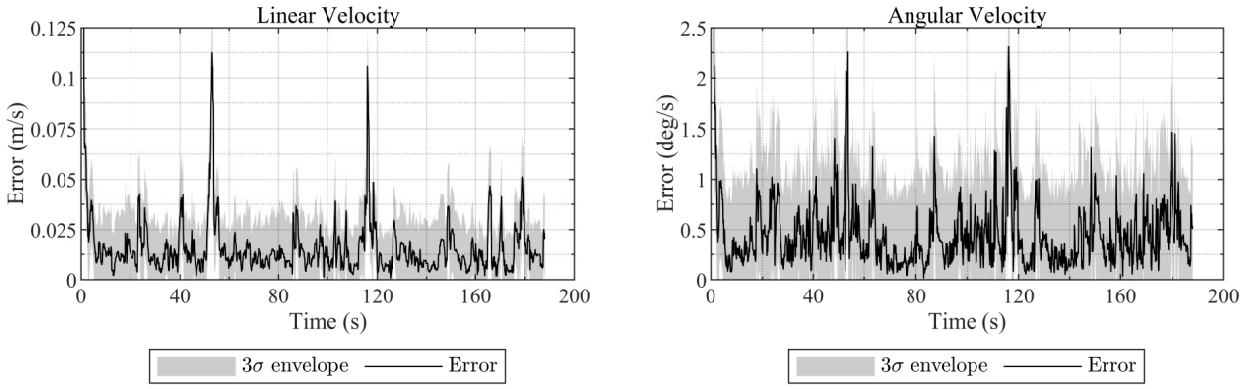


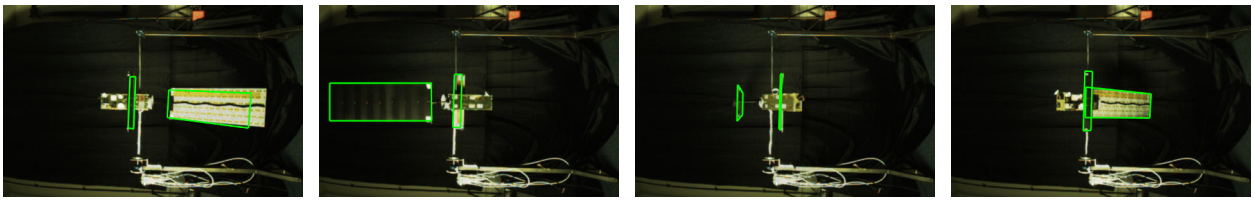
Fig. 24 Velocity estimation errors for the UASL dataset.

3. Validation on SPEED Dataset

Despite the analysis on the synthetic train dataset (cf. Section VIII.A.2), it would be interesting to obtain a more direct comparison with the state-of-the-art by assessing the pose estimation performance of the proposed method on the SPEED test data through the score obtained from the SPEC website. The SPEC score is computed as:

$$e_{\text{SPEC}} = \frac{1}{N}\sum_{i=1}^{N}\frac{e_{\text{pos},i}}{\|\boldsymbol{t}_{C/T,i}\|_2} + e_{\text{att},i}, \tag{55}$$

where N is the number of images in the test set. The score is computed automatically for both SPEED/TEST and SPEED/REAL-TEST, although only the former was used to decide the winners of the competition; the latter was shown for reference and to evaluate the transferability of the algorithm to laboratory data. During the competition, submissions were evaluated on a subset of all test images (also undisclosed) in order to avoid overfitting. At the end of the competition, the submissions were re-evaluated on the complete test sets and ranked accordingly. It is still possible to obtain a score on this subset by submitting the estimated 6-DOF values on the SPEC website. Since SPEED/TEST



(a) Initial pose at $t = 0$ s **(b)** $t = 37$ s **(c)** $t = 87$ s **(d)** $t = 137$ s

Fig. 25 Results of the relative pose estimation for the UASL dataset. The edges of the SAR and solar panel are reprojected in green using the estimated pose.

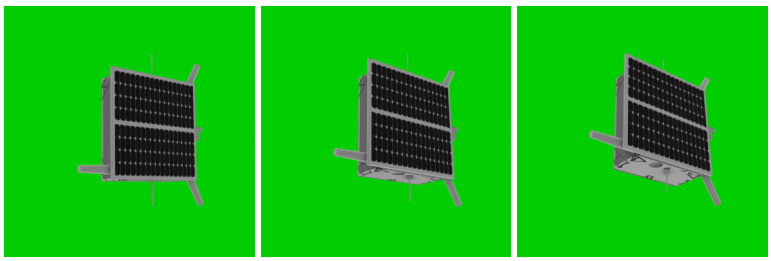
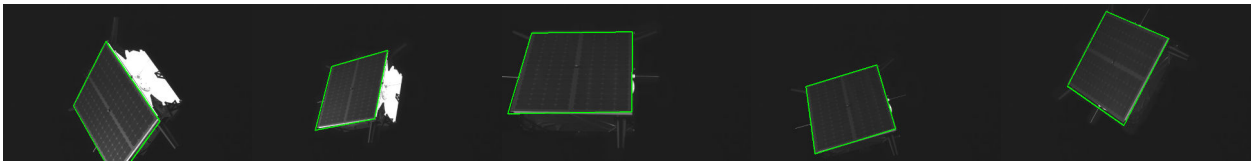


Fig. 26 Sample keyframes rendered from the reconstructed target spacecraft of SPEED.

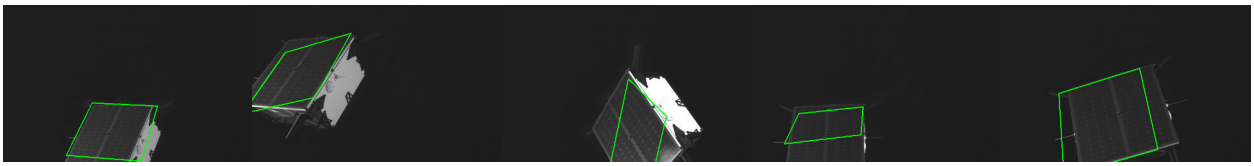
also contains images having the Earth in the background, which the present algorithm does not tackle, the performance is assessed for SPEED/REAL-TEST alone (i.e. the "real image score").

The CAD model of Tango has not been provided for SPEC. Therefore, for this section, the 3D structure of the spacecraft was first reconstructed using a few selected images from SPEED/TEST using multi-view triangulation from manually selected keypoints and the provided 6-DOF relative pose [71]. This was achieved using the Matlab Computer Vision Toolbox, yielding the corresponding set of 3D structural points, which were then imported to Blender and used as the reference to model Tango's geometric primitives and to texture the object. The reconstructed model was then used to render a number of keyframes covering the attitude range of the train set, SPEED/REAL. Figure 26 illustrates some sample keyframes rendered from this reconstructed model. As the resulting viewsphere is much more reduced compared to the evaluations in Section VIII.A, values of $\Delta^{\text{az}}_{\text{class}} = \Delta^{\text{el}}_{\text{class}} = 5$ deg are used instead (cf. Table 1), yielding a total of 12 possible classes for coarse pose determination.

The complete framework achieves a real image score $e_{\text{SPEC}} = 0.2692$. A qualitative illustration of the results can be observed in Fig. 27. Despite not having a background, the images from SPEED/REAL-TEST were found to contain some artefacts that made a typically straightforward threshold-based segmentation sub-optimal. Nonetheless, the algorithm is shown to generate a robust estimate of the relative pose (Fig. 27a). In some cases, the algorithm converges to a local minimum due to the target being partially outside of the FOV, which affects either the performance of the coarse module or the fine module, or both (Fig. 27b). However, these are a minority, and the attained e_{SPEC} is well below the Pytorch and Keras baseline scores of 2.6636 and 3.5359, respectively, provided by the SPEC organisers using deep learning.



(a) Optimal fits



(b) Sub-optimal fits

Fig. 27 Results of the relative pose estimation for the SPEED/REAL-TEST subset. The edges of the solar panel are reprojected in green using the estimated pose.

Table 9 Achieved SPEED/REAL-TEST subset e_{SPEC} score in the context of the scores obtained by the SPEC top-5 rankers in this metric.

Method	Username	Score	SPEC Final Rank
EPFL	`EPFL_cvlab`	0.1040	2
University of Surrey	`pedro_fairspace`	0.1476	3
Proposed	`dr_uasl`	0.2692	N/A
Stanford University	`stanford_slab`	0.3221	4
University of Adelaide	`UniAdelaide`	0.3634	1
Motoki Kimura	`motokimura1`	0.5714	6

The obtained score is set side-by-side to those achieved by the top-5 SPEC participants on the same dataset in Table 9; details about the methods can be found in Ref. [54]. It is clear that the proposed framework is comparable to the best scores on the SPEED/REAL-TEST subset, ranking third place. Notably, it is better than that achieved by the University of Adelaide (0.3634), the winners of the competition based on their SPEED/TEST score. It is reiterated that only synthetic data has been used for training, which demonstrates the robustness of the algorithm to the domain gap. For context, note that four out of the five competitor methods illustrated in Table 9 are confirmedly based on deep learning frameworks. Additionally, teams `UniAdelaide`, `EPFL_cvlab`, and `stanford_slab` all rely also on the ground truth 3D structural points of the target, which are used to train neural networks that detect their 2D coordinates in images; the actual pose is solved using PnP. At the time of submission, the result ranked fourth place on the post-mortem leaderboard in terms of real image score, although no details about the competing submissions are known.

IX. Conclusion

In this work, a robust, innovative model-based solution for spacecraft relative navigation using a single visible wavelength camera has been developed. The proposed contribution stands on the fact that the relative navigation solution is achieved using a set of discrete keyframes rendered in an offline stage from a three-dimensional model of the target spacecraft, where a 3D-2D problem is converted into a 2D-2D approach relying only on computer vision

methods capable of running exclusively on the CPU. The proposed method was validated using synthetic datasets closely simulating the imaging conditions experienced in-orbit, including scale changes and tumbling motion of the target, and a real dataset generated in laboratory featuring a (relative) circular observation trajectory about the complex spacecraft Envisat. The aspect of the target in each keyframe is learned using shape features and GMMs that are used to train a Bayesian classifier; the coarse viewpoint as seen by the camera was identified approximately 90% of the time with an error under 20 deg on the BLENDER dataset. The classifier was also tested on synthetic images from the open-source SPEED dataset, where it was found that a combination of sensor noise and low lighting conditions negatively affect the target shape extraction, bringing the performance down to 75% for 20 deg bounds.

The pose estimate is refined by matching hybrid features between the current image and train keyframe. Automatic outlier rejection is assured via M-estimation. An EKF is employed to fuse the pose hypotheses generated by each feature type; it is designed to operate on the tangent space of $SE(3)$, allowing it to seamlessly integrate the previous stage by assimilating the covariance matrices generated by the M-estimation directly as the measurement noise, independently of the pose parameterization. The attained solutions for both synthetic and laboratory datasets targeting Envisat showcase rapid convergence and yield a maximum error of 5% of the range distance in position and 3.8 deg in attitude; on average, steady-state errors are observed in the order of 2.5% of the range in position and 1 deg in attitude. A novel matching strategy by predicting feature locations using the EKF, alongside a RMSE-based validation gate, assures the stability and accuracy of the solution is maintained, even in the face of highly discrepant frames with respect to the database keyframes caused by light-scattering MLI and solar panel reflections.

Lastly, the proposed pipeline was validated on laboratory test images of SPEED, obtaining a pose score calculation of 0.2693, which demonstrates its robustness in bridging the domain gap between synthetic and real data and is comparable to the best scores obtained in the SPEC competition with deep learning. While the advances in the latter field towards computer vision tasks such as image classification are undeniable, the results achieved herein cast doubts on the belief that any deep learning-based technique is automatically capable of achieving a lower error than classical spacecraft relative pose estimation methods. It is noted, though, that the method is dependent on a proper extraction of the shape of the target and hence the score for synthetic images featuring Earth in the background could not be computed, something that can be easily surpassed using deep learning. Future work will thus focus on the development of a dedicated segmentation module to enhance the framework.

Acknowledgements

This work has been partially funded by Thales Alenia Space France (contract no. 1520056105). The authors would like to thank Dr Olivier Dubois-Matra from ESA for providing them access to the Astos Camera Simulator (contract no. 4000117583/16/NL/HK/as). The authors would also like to thank the anonymous reviewers for their constructive criticism.

References

[1] Bhaskaran, S., Desai, S., Dumont, P., Kennedy, B., Null, G., Owen Jr, W., Riedel, J., Synnott, S., and Werner, R., "Orbit Determination Performance Evaluation of the Deep Space 1 Autonomous Navigation System," *AAS/AIAA Space Flight Mechanics Meeting*, Monterey, CA, 1998.

[2] Castellini, F., Antal-Wokes, D., de Santayana, R. P., and Vantournhout, K., "Far Approach Optical Navigation and Comet Photometry for the Rosetta Mission," *Proceedings of the 25th International Symposium on Space Flight Dynamics*, Munich, Germany, 2015.

[3] Bercovici, B., and McMahon, J. W., "Point-Cloud Processing Using Modified Rodrigues Parameters for Relative Navigation," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 12, 2017, pp. 3167–3179. doi:10.2514/1.g002787.

[4] Bercovici, B., and McMahon, J. W., "Robust Autonomous Small-Body Shape Reconstruction and Relative Navigation Using Range Images," *Journal of Guidance, Control, and Dynamics*, Vol. 42, No. 7, 2019, pp. 1473–1488. doi:10.2514/1.g003898.

[5] Christian, J., Hinkel, H., Maguire, S., D'Souza, C., and Patangan, M., "The Sensor Test for Orion RelNav Risk Mitigation (STORRM) Development Test Objective," *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, 2011, p. 6260. doi:10.2514/6.2011-6260.

[6] Kechagias-Stamatis, O., Aouf, N., and Richardson, M., "High-speed multi-dimensional relative navigation for uncooperative space objects," *Acta Astronautica*, Vol. 160, 2019, pp. 388–400. doi:10.1016/j.actaastro.2019.04.050.

[7] Karacalıoğlu, A. G., and Stupl, J., "The Impact of New Trends in Satellite Launches on the Orbital Debris Environment," *8th IAASS Conference: Safety First, Safety for All*, NASA Ames Research Center, Melbourne, FL, United States, 2016. URL https://ntrs.nasa.gov/search.jsp?R=20160011184.

[8] Biesbroek, R., Innocenti, L., Wolahan, A., and Serrano, S., "e.Deorbit–ESA's Active Debris Removal Mission," *7th European Conference on Space Debris*, ESA Space Debris Office, 2017.

[9] Biesbroek, R., Wolahan, A., and Serrano, S. M., "e.Inspector: Clean Space Industrial Days," https://indico.esa.int/event/181/contributions/1378/attachments/1305/1530/e.Inspector_SARA.pdf, 2017. [Online; accessed June 2020].

[10] Tweddle, B. E., Saenz-Otero, A., Leonard, J. J., and Miller, D. W., "Factor Graph Modeling of Rigid-body Dynamics for Localization, Mapping, and Parameter Estimation of a Spinning Object in Space," *Journal of Field Robotics*, Vol. 32, No. 6, 2014, pp. 897–933. doi:10.1002/rob.21548.

[11] Yılmaz, O., Aouf, N., Majewski, L., and Sanchez-Gestido, M., "Using Infrared Based Relative Navigation for Active Debris Removal," *10th International ESA Conference on Guidance, Navigation and Control Systems*, Salzburg, Austria, 2017, pp. 1–16. URL http://dspace.lib.cranfield.ac.uk/handle/1826/12079.

[12] Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, Vol. 60, No. 2, 2004, pp. 91–110. doi:10.1023/b:visi.0000029664.99615.94.

[13] Rublee, E., Rabaud, V., Konolige, K., and Bradski, G., "ORB: An Efficient Alternative to SIFT or SURF," *2011 International Conference on Computer Vision*, IEEE, 2011, pp. 2564–2571. doi:10.1109/iccv.2011.6126544.

[14] Rondao, D., Aouf, N., Richardson, M. A., and Dubois-Matra, O., "Benchmarking of local feature detectors and descriptors for multispectral relative navigation in space," *Acta Astronautica*, Vol. 172, 2020, pp. 100–122. doi:10.1016/j.actaastro.2020.03.049.

[15] Lowe, D. G., "Three-Dimensional Object Recognition from Single Two-Dimensional Images," *Artificial Intelligence*, Vol. 31, No. 3, 1987, pp. 355–395. doi:10.1016/0004-3702(87)90070-1.

[16] Dhome, M., Richetin, M., Lapreste, J.-T., and Rives, G., "Determination of the Attitude of 3D Objects from a Single Perspective View," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 12, 1989, pp. 1265–1278. doi:10.1109/34.41365.

[17] "The vSLAM Algorithm for Robust Localization and Mapping," *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2005, pp. 24–29. doi:10.1109/robot.2005.1570091.

[18] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D., "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, Vol. 31, No. 5, 2015, pp. 1147–1163. doi:10.1109/tro.2015.2463671.

[19] Engel, J., Schöps, T., and Cremers, D., "LSD-SLAM: Large-Scale Direct Monocular SLAM," *Computer Vision – ECCV 2014*, Springer International Publishing, 2014, pp. 834–849. doi:10.1007/978-3-319-10605-2_54.

[20] Augenstein, S., and Rock, S., "Simultaneous Estimation of Target Pose and 3-D Shape Using the FastSLAM Algorithm," *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, 2009. doi:10.2514/6.2009-5782.

[21] Dor, M., and Tsiotras, P., "ORB-SLAM Applied to Spacecraft Non-Cooperative Rendezvous," *2018 Space Flight Mechanics Meeting*, American Institute of Aeronautics and Astronautics, 2018. doi:10.2514/6.2018-1963.

[22] Lepetit, V., and Fua, P., "Monocular Model-Based 3D Tracking of Rigid Objects: A Survey," *Foundations and Trends in Computer Graphics and Vision*, Vol. 1, No. 1, 2005, pp. 1–89. doi:10.1561/0600000001.

[23] Comport, A., Marchand, E., Pressigout, M., and Chaumette, F., "Real-Time Markerless Tracking for Augmented Reality: the Virtual Visual Servoing Framework," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 12, No. 4, 2006, pp. 615–628. doi:10.1109/tvcg.2006.78.

[24] Kelsey, J., Byrne, J., Cosgrove, M., Seereeram, S., and Mehra, R., "Vision-Based Relative Pose Estimation for Autonomous Rendezvous And Docking," *2006 IEEE Aerospace Conference*, IEEE, 2006. doi:10.1109/aero.2006.1655916.

[25] Petit, A., Marchand, E., and Kanani, K., "A Robust Model-Based Tracker Combining Geometrical and Color Edge Information," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2013. doi:10.1109/iros.2013.6696887.

[26] Zou, Y., Wang, X., Zhang, T., and Song, J., "Combining Point and Edge for Satellite Pose Tracking Under Illumination Varying," *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, IEEE, 2016. doi:10.1109/wcica.2016.7578814.

[27] Oumer, N. W., "Monocular 3D Pose Tracking of a Specular Object," *Proceedings of the 9th International Conference on Computer Vision Theory and Applications*, SCITEPRESS - Science and and Technology Publications, 2014. doi:10.5220/0004667304580465.

[28] Cai, J., Huang, P., Zhang, B., and Wang, D., "A TSR Visual Servoing System Based on a Novel Dynamic Template Matching Method," *Sensors*, Vol. 15, No. 12, 2015, pp. 32152–32167. doi:10.3390/s151229884.

[29] Gansmann, M., Mongrard, O., and Ankersen, F., "3D Model-Based Relative Pose Estimation for Rendezvous and Docking Using Edge Features," *10^{th} International ESA Conference on Guidance, Navigation and Control Systems*, 2017.

[30] Vacchetti, L., Lepetit, V., and Fua, P., "Fusing Online and Offline Information for Stable 3D Tracking in Real-Time," *Computer Society Conference on Computer Vision and Pattern Recognition Proceedings*, Vol. 2, IEEE, 2003, pp. II–241.

[31] Cropp, A., "Pose Estimation and Relative Orbit Determination of a Nearby Target Microsatellite using Passive Imagery," Ph.D. thesis, University of Surrey, United Kingdom, 2001. URL http://epubs.surrey.ac.uk/843875/.

[32] Lowe, D., "Fitting Parameterized Three-Dimensional Models to Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 5, 1991, pp. 441–450. doi:10.1109/34.134043.

[33] Abdenrahim, M., Diaz, J., Rossi, C., and Salichs, M., "Experimental Simulation of Satellite Relative Navigation Using Computer Vision," *Proceedings of 2nd International Conference on Recent Advances in Space Technologies (RAST)*, IEEE, 2005. doi:10.1109/rast.2005.1512596.

[34] Liu, C., and Hu, W., "Relative Pose Estimation for Cylinder-Shaped Spacecrafts using Single Image," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 50, No. 4, 2014, pp. 3036–3056. doi:10.1109/taes.2014.120757.

[35] Shi, J., Ulrich, S., and Ruel, S., "Spacecraft Pose Estimation Using a Monocular Camera," *67^{th} International Astronautical Congress*, Guadalajara, Mexico, 2016.

[36] Post, M., and Li, J., "Visual Monocular 3D Reconstruction and Component Identification for Small Spacecraft," 2018. doi:10.20944/preprints201801.0195.v1, Multidisciplinary Digital Publishing Institute (preprint).

[37] Kechagias-Stamatis, O., and Aouf, N., "Histogram of Distances for Local Surface Description," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016. doi:10.1109/icra.2016.7487402.

[38] Rondao, D., and Aouf, N., "Multi-View Monocular Pose Estimation for Spacecraft Relative Navigation," *2018 AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, 2018. doi:10.2514/6.2018-2100.

[39] Dudani, S. A., Breeding, K. J., and McGhee, R. B., "Aircraft Identification by Moment Invariants," *IEEE Transactions on Computers*, Vol. C-26, No. 1, 1977, pp. 39–46. doi:10.1109/tc.1977.5009272.

[40] Reeves, A., Prokop, R., Andrews, S., and Kuhl, F., "Three-Dimensional Shape Analysis Using Moments and Fourier Descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 6, 1988, pp. 937–943. doi:10.1109/34.9115.

[41] Breuers, M. G. J., "Image-based Aircraft Pose Estimation using Moment Invariants," *Automatic Target Recognition IX*, edited by F. A. Sadjadi, SPIE, 1999. doi:10.1117/12.359963.

[42] Ozuysal, M., Lepetit, V., and Fua, P., "Pose Estimation for Category Specific Multiview Object Localization," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009. doi:10.1109/cvpr.2009.5206633.

[43] Glasner, D., Galun, M., Alpert, S., Basri, R., and Shakhnarovich, G., "Viewpoint-Aware Object Detection and Pose Estimation," *2011 International Conference on Computer Vision*, IEEE, 2011. doi:10.1109/iccv.2011.6126379.

[44] Mei, L., Sun, M., Carter, K. M., III, A. O. H., and Savarese, S., "Unsupervised Object Pose Classification from Short Video Sequences," *Procedings of the British Machine Vision Conference 2009*, British Machine Vision Association, 2009. doi:10.5244/c.23.89.

[45] Kanani, K., Petit, A., Marchand, E., Chabot, T., and Gerber, B., "Vision Based Navigation for Debris Removal Missions," *63rd International Astronautical Congress*, Naples,Italy, 2012.

[46] Zhang, Y., Liu, H., and Shang, Y., "3D Model-based Detection and Tracking for Space Autonomous and Uncooperative Rendezvous," *AOPC 2015: Optical Design and Manufacturing Technologies*, edited by L. Li, L. Zheng, and K. P. Thompson, SPIE, 2015. doi:10.1117/12.2224964.

[47] Shi, J.-F., and Ulrich, S., "SoftPOSIT Enhancements for Monocular Camera Spacecraft Pose Estimation," *2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR)*, IEEE, 2016. doi:10.1109/mmar.2016.7575083.

[48] Shi, J.-F., Ulrich, S., and Ruel, S., "Spacecraft Pose Estimation using Principal Component Analysis and a Monocular Camera," *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, 2017. doi:10.2514/6.2017-1034.

[49] Sharma, S., Ventura, J., and D'Amico, S., "Robust Model-Based Monocular Pose Initialization for Noncooperative Spacecraft Rendezvous," *Journal of Spacecraft and Rockets*, Vol. 55, No. 6, 2018, pp. 1414–1429. doi:10.2514/1.a34124.

[50] Su, H., Qi, C. R., Li, Y., and Guibas, L. J., "Render for CNN: Viewpoint Estimation in Images Using CNNs Trained with Rendered 3D Model Views," *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, 2015. doi:10.1109/iccv.2015.308.

[51] Tulsiani, S., and Malik, J., "Viewpoints and Keypoints," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2015. doi:10.1109/cvpr.2015.7298758.

[52] Sharma, S., and D'Amico, S., "Pose Estimation for Non-Cooperative Rendezvous Using Neural Networks," , 2019. ArXiv:1906.09868.

[53] Proenca, P. F., and Gao, Y., "Deep Learning for Spacecraft Pose Estimation from Photorealistic Rendering," , 2019. ArXiv:1907.04298.

[54] Kisantal, M., Sharma, S., Park, T. H., Izzo, D., Martens, M., and Amico, S. D., "Satellite Pose Estimation Challenge: Dataset, Competition Design and Results," *IEEE Transactions on Aerospace and Electronic Systems*, 2020, pp. 1–1. doi:10.1109/taes.2020.2989063.

[55] Chen, B., Cao, J., Parra, A., and Chin, T.-J., "Satellite Pose Estimation with Deep Landmark Regression and Nonlinear Pose Refinement," *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, IEEE, 2019. doi:10.1109/iccvw.2019.00343.

[56] Szeliski, R., *Computer Vision: Algorithms and Applications*, 1st ed., Springer-Verlag, London, UK, 2011, pp. 45–47, 284–286. doi:10.1007/978-1-84882-935-0.

[57] Murray, R. M., Sastry, S. S., and Zexiang, L., *A Mathematical Introduction to Robotic Manipulation*, 1st ed., CRC Press, Inc., Boca Raton, FL, USA, 1994, pp. 34–39, 41–42, 53–54. doi:10.1201/9781315136370.

[58] Stillwell, J., *Naive Lie Theory*, Springer, New York, NY, 2008, pp. 82, 98. doi:10.1007/978-0-387-78214-0.

[59] Gallier, J., *Geometric Methods and Applications: For Computer Science and Engineering*, 2nd ed., Springer, New York, NY, 2011, pp. 468, 504–505. doi:10.1007/978-1-4419-9961-0.

[60] Selig, J. M., "Lie Groups and Lie Algebras in Robotics," *Computational Noncommutative Algebra and Applications*, edited by J. Byrnes, Springer, Dordrecht, Netherlands, 2004, pp. 101–125. doi:10.1007/1-4020-2307-3_5.

[61] Absil, P.-A., Mahony, R., and Sepulchre, R., *Optimization Algorithms on Matrix Manifolds*, Princeton University Press, Princeton, NJ, 2008, pp. 54–56. doi:10.1515/9781400830244.

[62] Gallier, J., and Quaintance, J., "Differential Geometry and Lie Groups I: A Computational Perspective," University of Pennsylvania (book in progress), August 2019. URL http://www.cis.upenn.edu/~jean/gbooks/manif.html, p. 591.

[63] Markley, F. L., and Crassidis, J. L., *Fundamentals of Spacecraft Attitude Determination and Control*, Space Technology Library, Springer, New York, NY, 2014, p. 45. doi:10.1007/978-1-4939-0802-8.

[64] Shuster, M. D., "A Survey of Attitude Representations," *Journal of the Astronautical Sciences*, Vol. 41, No. 4, 1993, pp. 439–517.

[65] Flusser, J., Suk, T., and Zitová, B., *2D and 3D Image Analysis by Moments*, 1st ed., John Wiley & Sons, Ltd, 2016, pp. 47–48, 352–356. doi:10.1002/9781119039402.

[66] Kintner, E. C., "On the Mathematical Properties of the Zernike Polynomials," *Optica Acta: International Journal of Optics*, Vol. 23, No. 8, 1976, pp. 679–680.

[67] Duda, R. O., Hart, P. E., and Stork, D. G., *Pattern Classification*, 2nd ed., John Wiley & Sons, New York, NY, 2012, pp. 21, 124–125.

[68] Figueiredo, M., and Jain, A., "Unsupervised Learning of Finite Mixture Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 3, 2002, pp. 381–396. doi:10.1109/34.990138.

[69] Li, S., Lee, M.-C., and Pun, C.-M., "Complex Zernike Moments Features for Shape-Based Image Retrieval," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, Vol. 39, No. 1, 2009, pp. 227–237. doi:10.1109/tsmca.2008.2007988.

[70] Lepetit, V., Moreno-Noguer, F., and Fua, P., "EPnP: An Accurate O(n) Solution to the PnP Problem," *International Journal of Computer Vision*, Vol. 81, No. 2, 2008, pp. 155–166. doi:10.1007/s11263-008-0152-6.

[71] Hartley, R., and Zisserman, A., *Multiple View Geometry in Computer Vision*, 2nd ed., Cambridge University Press, Cambridge, UK, 2004, pp. 181, 135, 141–142, 434–435. doi:10.1017/cbo9780511811685.

[72] Kanatani, K., *Statistical Optimization for Geometric Computation: Theory and Practice*, Elsevier Science Inc., New York, NY, 1996, p. 67. doi:10.1016/s0923-0459(96)x8019-4.

[73] Rosten, E., and Drummond, T., "Machine Learning for High-Speed Corner Detection," *Computer Vision – ECCV 2006*, Springer Berlin Heidelberg, 2006, pp. 430–443. doi:10.1007/11744023_34.

[74] Rondao, D., Aouf, N., and Dubois-Matra, O., "Multispectral Image Processing for Navigation Using Low Performance Computing," *69th International Astronautical Congress (IAC) 2018*, IAF, Bremen, Germany, 2018. URL https://dspace.lib.cranfield.ac.uk/handle/1826/13558.

[75] Canny, J., "A Computational Approach to Edge Detection," *Readings in Computer Vision*, Elsevier, 1987, pp. 184–203. doi:10.1016/b978-0-08-051581-6.50024-6.

[76] Lee, J. H., Lee, S., Zhang, G., Lim, J., Chung, W. K., and Suh, I. H., "Outdoor Place Recognition in Urban Environments using Straight Lines," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014. doi:10.1109/icra.2014.6907675.

[77] Alahi, A., Ortiz, R., and Vandergheynst, P., "FREAK: Fast Retina Keypoint," *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 510–517. doi:10.1109/cvpr.2012.6247715.

[78] Markovsky, I., and Mahmoodi, S., "Least-Squares Contour Alignment," *IEEE Signal Processing Letters*, Vol. 16, No. 1, 2009, pp. 41–44. doi:10.1109/lsp.2008.2008588.

[79] Rousseeuw, P. J., and Leroy, A. M., *Robust Regression and Outlier Detection*, John Wiley & Sons, Inc., New York, NY, 1987, pp. 1–4, 12. doi:10.1002/0471725382.

[80] Zhang, Z., "Parameter Estimation Techniques: A Tutorial with Application to Conic Fitting," *Image and Vision Computing*, Vol. 15, No. 1, 1997, pp. 59–76. doi:10.1016/s0262-8856(96)01112-2.

[81] Holland, P. W., and Welsch, R. E., "Robust Regression Using Iteratively Reweighted Least-Squares," *Communications in Statistics - Theory and Methods*, Vol. 6, No. 9, 1977, pp. 813–827. doi:10.1080/03610927708827533.

[82] Huber, P., "Robust Methods of Estimation of Regression Coefficients," *Series Statistics*, Vol. 8, No. 1, 1977, pp. 41–53. doi:10.1080/02331887708801356.

[83] Beaton, A. E., and Tukey, J. W., "The Fitting of Power Series, Meaning Polynomials, Illustrated on Band-Spectroscopic Data," *Technometrics*, Vol. 16, No. 2, 1974, pp. 147–185. doi:10.1080/00401706.1974.10489171.

[84] Stewart, C. V., "Robust Parameter Estimation in Computer Vision," *SIAM Review*, Vol. 41, No. 3, 1999, pp. 513–537. doi:10.1137/s0036144598345802.

[85] Huber, P. J., *Robust Statistics*, 2nd ed., John Wiley & Sons, Inc., 2009, pp. 175–186. doi:10.1002/0471725250.

[86] Barfoot, T. D., *State Estimation for Robotics*, 1st ed., Cambridge University Press, New York, NY, 2017, pp. 265, 359. doi:10.1017/9781316671528.

[87] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O., "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 6, 2007, pp. 1052–1067. doi:10.1109/tpami.2007.1049.

[88] Grewal, M., and Andrews, A., *Kalman Filtering: Theory and Practice Using MATLAB*, 4th ed., Wiley, Hoboken, NJ, 2015, p. 136.

[89] Blanco, José-Luis, "A Tutorial on SE(3) Transformation Parameterizations and On-Manifold Optimization," Tech. rep., University of Málaga, Oct. 2018. URL https://w3.ual.es/~jlblanco/publications/#publications.

[90] Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C., "Visual Categorization with Bags of Keypoints," *8th European Conference on Computer Vision (ECCV)*, Vol. 1, Prague, Czech Republic, 2004, pp. 1–2.

[91] Zhang, J., Marszalek, M., Lazebnik, S., and Schmid, C., "Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study," *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, IEEE, 2006. doi:10.1109/cvprw.2006.121.